

UNIVERSITÉ DE SHERBROOKE
Faculté de génie
Département de génie électrique et de génie informatique

INTERPRÉTATION VISUELLE DE
SYMBOLES PAR UN ROBOT MOBILE

Mémoire de maîtrise ès sciences appliquées
Spécialité : génie électrique

Dominic LÉTOURNEAU

Sherbrooke (Québec), Canada

Décembre 2001

SOMMAIRE

Pour un robot mobile, être capable de lire des symboles dont la signification est partagée avec les humains est primordiale pour que ces machines puissent évoluer dans des environnements courants. Doter les robots d'une telle capacité est très utile pour leur localisation à partir d'insignes ou pour le repérage d'objets ou de lieux importants. Les travaux effectués dans ce mémoire visent à développer les mécanismes et les algorithmes requis pour la reconnaissance des symboles à partir d'une caméra couleur. Les étapes pour y arriver sont l'extraction du symbole dans l'image, le contrôle du robot et de la caméra pour l'acquisition d'une image adéquate du symbole à identifier, et la reconnaissance du symbole. La segmentation des couleurs pour l'extraction du symbole dans l'image, l'approche comportementale hybride pour le contrôle du robot et de la caméra et l'utilisation d'un réseau de neurones à propagation avant pour la reconnaissance des symboles permettent de parvenir au but fixé en respectant les différentes conditions d'opération. La validation des travaux s'effectue à partir de 25 symboles pré-sélectionnés (lettres, chiffres, signes) imprimés en noir sur des feuilles 8.5" x 11" de trois couleurs différentes (bleu, rose, orange) et sous deux niveaux d'éclairage. Les résultats obtenus démontrent que les méthodes proposées permettent à un robot mobile d'interpréter un symbole en environ dix secondes à une distance moyenne de cinq pieds, et avec un taux de reconnaissance moyen d'environ 90%. Ces travaux démontrent la faisabilité de faire lire des symboles par un robot mobile et servent de point de départ pour des recherches visant à faire lire par des robots des messages de plus en plus complexes, comprenant plusieurs symboles agencés pour former des mots ou des phrases.

REMERCIEMENTS

D'abord, je voudrais remercier le professeur François Michaud qui m'a permis de réaliser ce projet. Ses conseils précieux, son expérience en recherche et surtout son efficacité incroyable m'ont permis de réaliser ce projet d'envergure en un temps record. Je souhaite que tous les étudiants puissent avoir un directeur de maîtrise du calibre du professeur Michaud.

Je voudrais également remercier toutes les personnes qui ont contribué au succès de ce projet : Jonathan Audet, Jean-Marc Valin, Luc Lussier, Catherine Théberge-Turmel, Serge Caron et Joey Fugère. Les nombreuses heures de discussion et de développement en équipe ont permis la réalisation rapide de ce projet et l'élaboration d'autres projets très ambitieux pour l'avenir.

Je voudrais aussi formuler des remerciements spéciaux à toute ma famille ainsi qu'à Linda Vân-Anh Truong et sa famille. Toutes ces personnes m'ont appuyé tout au long de mon cheminement aux études graduées et m'ont permis de me dépasser de jour en jour.

Finalement, je tiens à remercier tous les autres membres du groupe de recherche LABORIUS ainsi que les professeurs et amis qui ont contribué à mon projet par leurs opinions et leurs commentaires.

TABLE DES MATIÈRES

1	INTRODUCTION	1
2	CARACTÉRISTIQUES DU MATÉRIEL D'EXPÉRIMENTATION	9
2.1	Spécifications techniques du robot mobile utilisé	9
2.2	Représentation des symboles.....	13
3	EXTRACTION DU SYMBOLE.....	15
3.1	Indices visuels pour l'extraction d'un symbole dans une image	15
3.1.1	Perception de la couleur	15
3.1.2	Détection de contours	17
3.1.3	Perception des textures	18
3.1.4	Choix de la méthode pour l'extraction du symbole.....	18
3.2	Représentation des couleurs	18
3.3	Algorithme de perception des couleurs pour les symboles	22
3.4	Algorithme de segmentation des couleurs.....	27
3.5	Analyse des performances pour l'extraction des couleurs	29
4	POSITIONNEMENT DU ROBOT ET DE LA CAMÉRA	31
4.1	Contrôle des déplacements du robot.....	31
4.2	Positionnement de la caméra	33
5	RECONNAISSANCE DU SYMBOLE	41
5.1	Réseaux de neurone à propagation avant avec apprentissage par rétro- propagation	41
5.2	Représentation des symboles.....	45
5.2.1	Le dimensionnement (<i>scaling</i>)	45
5.2.2	La DCT	46

5.3 Performances de reconnaissance de symboles	49
6 RÉSULTATS.....	53
6.1 Expérimentations avec éclairage contrôlé	53
6.2 Expérimentations dans des conditions d'éclairage non-contrôlées	59
7 DISCUSSION.....	61
CONCLUSION	64
BIBLIOGRAPHIE.....	67
ANNEXE 1 - MATRICE DE CONFUSION POUR LES SYMBOLES ENCODÉS PAR DIMENSIONNEMENT	71
ANNEXE 2 - MATRICE DE CONFUSION POUR LES SYMBOLES ENCODÉS PAR DCT	72

LISTE DES FIGURES

Figure 2.1 Robot Pioneer 2.....	10
Figure 2.2 Architecture matérielle d'un robot Pioneer 2.....	12
Figure 2.3 Robot Pioneer 2 en face d'un symbole	14
Figure 3.1 Espace de couleur RGB avec une région définie par seuils constants.....	16
Figure 3.2 Espace de couleur HSV.....	21
Figure 3.3 Table d'appartenance des couleurs	24
Figure 3.4 Interface-usager pour l'ajustement des couleurs (par exemple le noir)	26
Figure 3.5 Représentation dans l'espace RGB des quatre couleurs utilisées	27
Figure 3.6 Algorithme de connexion des cellules de même couleur.....	29
Figure 4.1 Organisation des modules décisionnels du robot.....	32
Figure 4.2 Images a) avant et b) après le positionnement de la caméra	34
Figure 4.3 Algorithme pour centrer le symbole	35
Figure 4.4 Facteur d'agrandissement de la caméra SONY EVI-D30.....	37
Figure 5.1 Schéma d'un neurone artificiel	42
Figure 5.2 Réseau de neurones à propagation avant avec une couche cachée	43
Figure 5.3 Dimensionnement du caractère E	45
Figure 5.4 Représentation graphique de la transformée en cosinus (DCT).....	47
Figure 5.5 Reconstitution d'un symbole à partir des coefficients de basses fréquences de dimension 9 x 9	48
Figure 7.1 Identification d'une station de recharge par un symbole	61

LISTE DES TABLEAUX

TABLEAU 2.1 Spécifications de la caméra <i>Sony</i> EVI-D30	11
TABLEAU 4.1 Temps de capture d'un symbole à différentes distances.....	40
TABLEAU 5.1 Évaluation de configurations de réseaux encodés par dimensionnement...	50
TABLEAU 5.2 Évaluation de configurations de réseaux encodés pour la DCT	51
TABLEAU 6.1 Taux de reconnaissance pour l'encodage par dimensionnement.....	54
TABLEAU 6.2 Taux de reconnaissance pour l'encodage par DCT	55
TABLEAU 6.3 Taux de reconnaissance de chacun des symboles.....	56
TABLEAU 6.4 Taux de reconnaissance des symboles pour différentes résolutions.....	58
TABLEAU 6.5 Temps de capture optimisés à différentes distances	59

1 INTRODUCTION

L'utilisation de robots mobiles dans la vie de tous les jours demande de concevoir des machines capables d'interagir avec différents intervenants, souvent qualifiés d'agents dans la terminologie utilisée en intelligence artificielle, sous différentes formes et pour des fins variées. Pour qu'un robot mobile soit utile et bien adapté aux situations pouvant être rencontrées dans la vie courante, il se doit de faire preuve d'une certaine intelligence sociale et d'évoluer dans un milieu dynamique.

Parmi les habiletés requises en robotique sociale [DAUTENHAHN 1999], deux semblent plus fondamentales pour mettre en place des capacités d'interaction efficaces entre agents :

- Les agents peuvent se reconnaître, interagir entre eux et participer à des interactions sociales.
- Les agents peuvent communiquer de manière explicite entre eux. La signification des informations communiquées est déterminée par un contexte partagé entre les interlocuteurs.

Pour les être vivants, certains sens permettent de rencontrer ces deux habiletés simultanément. Par exemple, les cris d'animaux ou la parole permettent de communiquer des messages. La vision permet aussi d'identifier un interlocuteur et de communiquer des informations par des gestes ou encore par des expressions corporelles ou faciales.

Pour les agents, la communication s'effectue sous trois principales formes [CAO et coll. 1997]: par l'environnement, par les sens et par l'envoi explicite de signaux.

La communication par l'environnement est la plus simple et la plus limitée des trois formes. Elle utilise l'environnement en soi comme médium de communication. Il n'y a pas de communication explicite ou d'interaction directe entre les agents. Par exemple, les fourmis communiquent la position de la nourriture en laissant des traces de phéromones dans l'environnement en partant du nid jusqu'à la nourriture. En suivant cette trace, les fourmis qui n'avaient jamais utilisé la route auparavant peuvent facilement retrouver leur chemin jusqu'à la nourriture [RESNIK 1991]. Cette forme de communication est communément appelée « la coopération sans communication » [ARKIN 1993][SEN 1994]. L'environnement se trouve ainsi l'équivalent d'une mémoire partagée pour les différents agents. Plusieurs systèmes ont déjà été développés en utilisant cette technique, souvent à partir des moyens de communication implicites des insectes [BECKERS et coll. 1994]. Le terme « stigmergie » est alors utilisé pour qualifier l'effet résultant du travail collectif d'agents qui se communiquent de l'information via l'environnement.

La communication par les sens est une forme plus évoluée et réfère à l'interaction locale qui se produit entre deux agents lorsqu'ils se détectent par leurs senseurs, mais toujours sans échange explicite d'information. Ce type de communication demande que les agents aient la capacité de se distinguer les uns des autres dans un groupe (qualifiée de *kin sensing* [MATARIC 1993][MATARIC 1994]) et de se distinguer par rapport aux objets de l'environnement. La communication via les sens est indispensable pour la modélisation des agents qui se trouvent dans un même environnement. À cause de limitations matérielles, ce moyen de communication est habituellement émulé en utilisant un dispositif de communication par infrarouge [MCLURKIN 2000] ou par une communication radio des

positions absolues des robots, obtenues à partir d'un système de positionnement commun [DUDEK et coll.1993].

L'identification des robots ou des objets dans l'environnement est souvent effectuée par la reconnaissance des couleurs. Dans la plupart des cas, l'identification s'effectue en attribuant une couleur prédéterminée aux objets à reconnaître [ARKIN 1998]. Par exemple, les robots d'une même équipe de soccer peuvent porter un dossard afin de s'identifier entre eux [VELOSO et coll. 2000]. Une séquence de couleurs est aussi utilisée pour identifier les autres robots qui se trouvent dans l'environnement [MAXWELL et coll. 2001]. Par contre, un robot de couleur déterminée peut facilement être confondu avec tout autre objet de la même couleur dans la pièce. Une approche intéressante consiste à utiliser des signaux lumineux pour communiquer des informations entre les robots [MICHAUD et coll. 1999]. L'utilisation d'un protocole pour la transmission des messages permet de discriminer un objet d'un autre intervenant. De plus, l'utilisation d'une source lumineuse rend la détection des couleurs plus robuste aux variations de luminosité. Son principal désavantage est le faible débit de transmission de l'information qui est dû aux capacités limitées de traitement temps réel des images successives. Cependant, l'utilité de la communication par signalement lumineux ne réside pas dans la vitesse de transmission, mais plutôt dans le partage du contexte perceptuel des agents en communication, évitant ainsi d'avoir à transmettre toutes les informations requises pour interagir entre eux (comme leur position relative, la présence d'un objet entre eux et qu'ils peuvent percevoir, etc.).

Finalement, la communication par l'envoi de signaux consiste à envoyer explicitement des messages aux autres agents. Ces messages peuvent être dirigés vers un agent récepteur ou bien transmis à tous les agents en même temps. Les agents doivent toutefois partager un

protocole de communication afin d'interagir convenablement entre eux. Le médium de communication influence alors le protocole de communication utilisé. Par exemple avec des médiums électroniques de communication (infrarouge, radio, Ethernet ou autres), ces formes de communications sont similaires à celles qui existent déjà pour les réseaux et les protocoles déjà développés sont réutilisés pour des robots mobiles. De grandes quantités d'informations peuvent alors être transmises en format numérique. Ces formes de communication nécessitent toutefois l'utilisation de dispositifs d'émission et de réception pour l'interprétation des signaux électroniques, et ne donnent aucune information sur le positionnement du robot dans son environnement. D'autres dispositifs électroniques pour le positionnement peuvent alors être utilisés (GPS ou un système de référence basé sur la triangulation d'ondes radio [EVERETT 1995]). Les informations obtenues par ces dispositifs peuvent alors être transmises par communication radio [MATARIC 1994][PARKER 1994].

En considérant ces trois formes de communication, il ressort qu'un mélange d'interaction par les sens et de communication explicite serait souhaitable pour réaliser une interaction sociale évoluée avec un robot mobile. De plus, la vision artificielle apparaît comme un moyen intéressant pour y arriver. En effet, la vision peut servir à percevoir d'autres intervenants ou des objets. Pour la communication, elle peut servir à interpréter des symboles et à agir selon leur sémantique (ce qui constitue le protocole de communication), comme peuvent le faire les intervenants humains et ce, sans dispositifs spéciaux. La vision s'avère une méthode plus naturelle et complémentaire à d'autres moyens électroniques de communication et c'est dans cette optique que ce mémoire étudie la question.

Avec les progrès technologiques des dernières années, il est maintenant possible de munir un robot d'une caméra couleur et d'utiliser des algorithmes de vision pour extraire des caractéristiques importantes de l'image (comme la couleur, les contours, les formes, etc.). Le tout doit toutefois s'effectuer en temps réel sur des systèmes embarqués. Il faut alors développer des moyens robustes et efficaces de récupérer les informations pertinentes dans l'image. Dans le but de permettre à un robot mobile d'extraire des informations de l'environnement, l'approche préconisée pour le mémoire consiste à développer pour un robot mobile la capacité d'interpréter visuellement des symboles.

Une telle habileté est certainement très utile dans notre société. Les humains utilisent des symboles pour véhiculer toutes sortes d'informations : numéros de chambres, noms sur les portes de bureaux, noms de rues, signalisation routière, etc. Les déplacements en automobile illustrent très bien l'utilité des symboles pour aider à nous localiser. Par exemple, pour aller de Sherbrooke à Montréal, nous n'avons qu'à consulter une carte routière pour obtenir une idée générale du trajet à suivre. Malgré l'existence d'une telle carte, nous avons tout de même besoin d'indications visuelles dérivées de symboles ou de signes pour confirmer notre position dans l'environnement. Il est impensable de mesurer les distances exactes à parcourir et baser notre navigation seulement sur les lectures prises à l'odomètre. Des erreurs dans les mesures vont très probablement survenir, ce qui rend la technique inadéquate. Toutefois, il nous est possible de se diriger dans la bonne direction en se référant aux panneaux de signalisation. Ceux-ci nous donnent des indices visuels indispensables que nous pouvons associer aux endroits stratégiques de la carte.

Les mêmes observations restent valides pour des robots mobiles. Les travaux portant sur la localisation de robots mobiles se basent principalement sur l'utilisation de cartes couplées à des senseurs de proximité (par exemple sonars, laser, etc.) [THRUN et coll. 1998]. Utiliser des repères visuels pourrait améliorer ces approches. Une solution consiste à extraire des caractéristiques visuelles uniques à l'environnement du robot [REKLEITIS et coll. 1995] à partir des images provenant de la caméra du robot. Dans ce cas, l'environnement est considéré comme fixe et le robot doit déjà connaître la position des repères afin d'y naviguer convenablement. Une autre possibilité, celle qui est utilisée dans ce mémoire, est plus générale. Elle permet au robot de reconnaître des symboles (lettres, dessins, etc.) qui sont disposés n'importe où dans l'environnement. Doter les robots d'une telle capacité est une idée intéressante puisque c'est une méthode qui peut être partagée par différents types de robots, en autant qu'ils possèdent un système de vision. Les symboles sont également interprétables par les humains, contrairement aux moyens électroniques de communication utilisés par les robots.

L'idée de faire lire les machines n'est pas nouvelle et les recherches sur ce sujet ont débutées il y a près de 40 ans [SUEN et coll. 1990]. Par exemple, en 1958 Frank Rosenblatt a construit l'ordinateur nommé Mark I capable de reconnaître des caractères à partir d'un réseau de neurones de type Perceptron [HECHT et coll. 1989]. Plus récemment, plusieurs produits commerciaux sont en mesure de reconnaître des caractères manuscrits. De ce fait, mettre en oeuvre la capacité de reconnaître des symboles sur un agent mobile autonome est sûrement un projet réalisable et qui n'a d'ailleurs pas été réalisé jusqu'à maintenant avec une caméra non fixe contrôlée intentionnellement par le robot. Cette problématique présente toutefois un défi important, car elle demande de mettre en place des moyens pour

percevoir le symbole, positionner le robot par rapport au symbole à interpréter, identifier le symbole et prendre des actions en conséquence. L'objectif du présent ouvrage est de développer toutes les habiletés nécessaires afin qu'un robot mobile autonome soit en mesure d'interpréter visuellement des symboles placés dans un environnement réel. Ainsi, le but n'est pas de concevoir de nouveaux algorithmes pour la reconnaissance de symboles ou encore de développer des dispositifs pour y arriver. Des approches existantes sont plutôt intégrées aux mécanismes requis pour permettre à un robot mobile ayant des capacités de traitement limitées d'identifier de façon autonome des symboles.

Ce mémoire est organisé en suivant la même séquence des opérations réalisées par l'approche développée pour l'interprétation visuelle de symbole pour un robot mobile. Pour débiter, le chapitre 2 expose les fonctionnalités de la plate-forme robotique utilisée avec sa caméra couleur, ainsi que les caractéristiques des symboles identifiables par le robot. Ensuite, les trois étapes de traitement requises pour l'identification de symbole par un robot mobile sont décrites. L'extraction de symbole dans une image est basé sur la segmentation de couleurs, sujet qui est abordé au chapitre 3. La technique utilisée pour la navigation du robot et le positionnement de la caméra pour l'acquisition d'une image du symbole à reconnaître est expliqué au chapitre 4. Une approche comportementale couplée avec un module d'interprétation est utilisée, avec un contrôleur PID pour le positionnement de la caméra Pan-Tilt-Zoom. Le chapitre 5 présente les réseaux de neurones de type propagation avant avec les entraînements réalisés par rétro-propagation pour la reconnaissance des symboles. Un des objectifs poursuivis lors de la mise en œuvre de ces trois étapes de traitement est d'arriver à utiliser des mécanismes les plus simples possibles afin de démontrer la faisabilité d'identifier des symboles par un robot mobile. Les performances

obtenues pour l'ensemble de ces étapes dans des conditions d'opération en environnement réel sont présentées au chapitre 6, suivi au chapitre 7 par une discussion sur les développements futurs rattachés au projet décrit dans cet ouvrage.

2 CARACTÉRISTIQUES DU MATÉRIEL D'EXPÉRIMENTATION

La plate-forme robotique avec son système de vision amènent certaines contraintes quant à la mise en œuvre en temps réel sur un robot mobile. Puisque l'objectif des travaux consiste à donner au robot la capacité de reconnaître des symboles dans des environnements réels, il est donc important de bien mettre en évidence les caractéristiques pouvant influencer l'atteinte de cet objectif. De plus, ce chapitre décrit les choix posés au niveau de la représentation des symboles dans le but d'identifier les problèmes fondamentaux rattachés à l'objectif de recherche du présent ouvrage.

2.1 Spécifications techniques du robot mobile utilisé

Le robot utilisé pour les expériences est un robot *Pioneer 2* de la compagnie *ActivMedia*, tel que montré à la figure 2.1. Le robot est équipé de seize sonars, huit à l'avant et huit à l'arrière. Ces sonars servent à détecter la présence d'objets à proximité du robot. Le robot possède également une boussole électronique, des pinces à l'avant pour saisir des objets et un modem Ethernet sans fil à haut débit (3 Mbps).

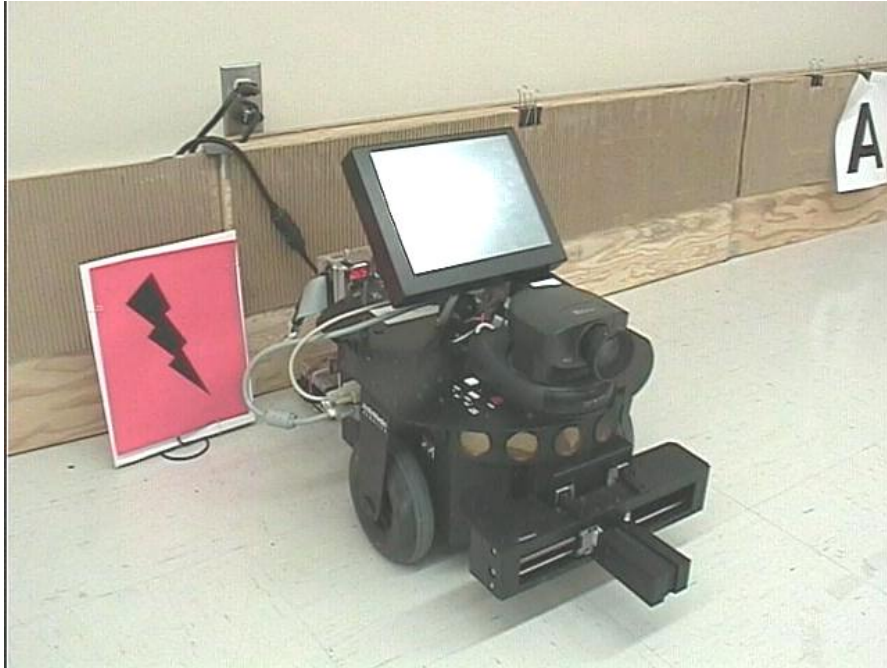


Figure 2.1 Robot Pioneer 2

Le système de vision du robot est composé d'une caméra *Sony EVI-D30* et d'une carte de capture vidéo *PXC200* fabriquée par la compagnie *Imagination inc.* Les spécifications techniques de la caméra sont présentées au tableau 2.1. La caméra permet des déplacements horizontaux (Pan), verticaux (Tilt) et un agrandissement optique (Zoom) de 12X. Une lentille grand angle est aussi installée sur la caméra. La lentille grand angle permet d'élargir le champ de vision en périphérie du robot et d'obtenir une meilleure perception des objets qui ne sont pas perceptibles avec les lentilles conventionnelles. La carte de capture *PXC200* peut fournir un maximum de trente images par seconde en mode continu (transfert direct en mémoire des images sans l'utilisation du processeur), en utilisant le format RGB (15,16 ou 24 bits) ou YUV (422 à 16 bits), avec une résolution maximale de 640 colonnes par 480 lignes. Étant donné que le traitement des images de dimension 640 x 480 n'est pas possible en temps réel sur l'ordinateur embarqué du robot, la dimension des images captées par la

carte de capture PXC200 est fixée à 320 x 240. Cette configuration permet le traitement de quatre fois moins de pixels et permet un traitement en temps réel.

TABEAU 2.1 Spécifications de la caméra *Sony* EVI-D30¹

Signal vidéo	NTSC
Senseur vidéo	1/3'' Hyper HAD CCD
Nombre de pixels	769(H) X 492(V)
Résolution horizontale	460 lignes (TV)
Résolution verticale	350 lignes (TV)
Lentille	Zoom optique 12X, f = 5.4 à 64.9 mm, F2.8 à F2.7
Angle de vue horizontal	4.3 à 48.4 degrés
Distance minimale du sujet	10mm (WIDE), 800mm (TELE)
Illumination	7 à 100000 lx
Exposition automatique	Iris automatique, AGC
Vitesse d'obturation	1/60 à 1/10000 (RS-232)
Gain	Automatique/manuel (RS-232)
Balance du blanc	ATW (one push hold), config. Intérieur/extérieur (RS-232)
Ratio Signal/Bruit	Plus de 48 dB
Pan/Tilt	Horizontal ±100 degrés (80 degrés/s maximum) Vertical ±25 degrés (50 degrés/s maximum) RS-232
Sortie Vidéo	RCA, 1 Vpp, 75 Ohm
Sortie S-Vidéo	4 broches mini DIN
Contrôle	RS-232C, 8 broches mini DIN, 9600bps, Data 8 bits, Stop 1 bit
Alimentation	DC 12 V à 14 V
Puissance	11 W
Température d'opération	0 à 40 °C
Dimensions	142 mm x 109 mm x 164 mm
Poids	1200 g
Accessoires	Télécommande infrarouge

De plus, l'absence de pilote sur Linux pour accéder aux images en mode continu demande que les images soient obtenues par requêtes successives à la carte PXC200. Le temps d'acquisition d'une image dépend ainsi de la vitesse du système d'exploitation à traiter la

¹ <http://www.sony.co.jp/en/Products/ISP/products/color/EVID30.html>

requête, au temps de communication entre le processeur et la carte PXC200 par le bus PCI et au temps d'envoi de l'image de la carte PXC200 à la mémoire désignée par le système d'exploitation. Le nombre d'images pouvant être traitées par seconde est donc limité approximativement à dix.

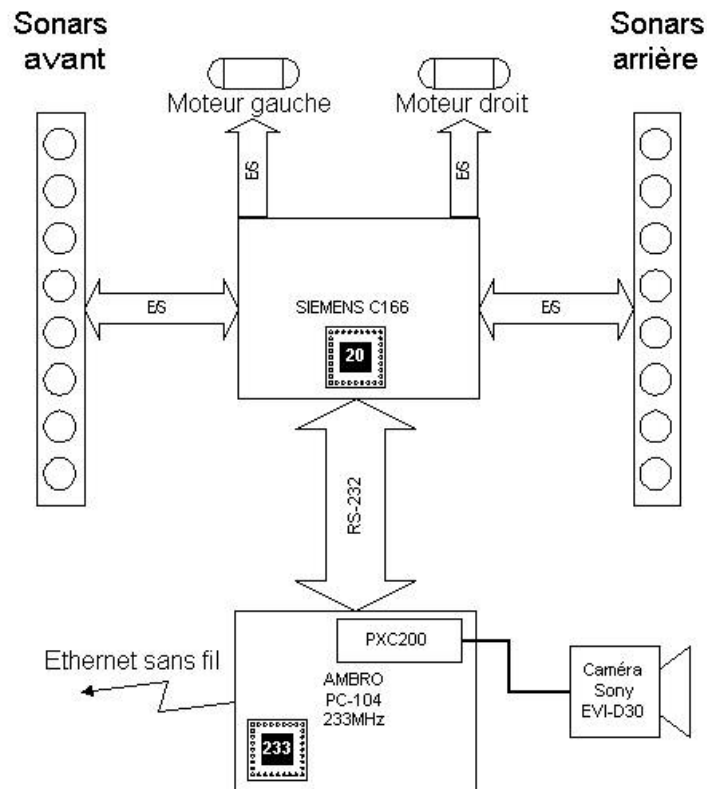


Figure 2.2 Architecture matérielle d'un robot Pioneer 2

L'architecture matérielle interne du robot est illustrée à la figure 2.2. Un ordinateur de type PC-104 de la compagnie *Ambro*, avec un processeur Pentium cadencé à 233 MHz, 32 Mo de mémoire vive et un disque dur de 3 Go, servent pour la prise de décision du robot. Le système d'exploitation est *RedHat Linux* version 6.2 et l'environnement de programmation du robot est *Ayllu* [WERGER 2000]. Un microcontrôleur de type *Siemens* C166 à 20 MHz sert d'interface aux capteurs et actionneurs du robot. La lecture des capteurs et l'envoi de commandes aux actionneurs s'effectue à une fréquence maximale de 20 Hz. Le temps de

réponse maximal possible est donc de 50 msec. Les échanges entre le microcontrôleur Siemens C166 et l'ordinateur Pentium 233 MHz se font via un lien série (RS-232) à un rythme de 9600 baud.

2.2 Représentation des symboles

Les symboles à identifier sont de police *Arial* de taille 650 points, imprimés en noir et centrés sur des feuilles de couleur 8.5" x 11". Des feuilles de couleur orange, bleu et rose sont utilisées. Ces couleurs de fond assurent une bonne démarcation du symbole avec les autres éléments noirs dans l'environnement. Aucun symbole n'entre en contact avec les bords de la feuille. De plus, les symboles sont disposés perpendiculairement au sol sur des surfaces planes, à la hauteur du robot. Ceci limite la distorsion de la perception du symbole pour différents angles de vision. La figure 2.3 montre un robot en face d'un symbole, prêt à procéder à son interprétation. Enfin, toujours dans le but de limiter la complexité de l'approche et d'assurer sa mise en œuvre en temps réel, les deux contraintes suivantes sont posées :

- seul le plus gros symbole dans l'image est considéré;
- chaque symbole est formé d'un seul segment, c'est-à-dire que tous les pixels de l'image qui composent le symbole sont connectés par au moins un voisin. Ceci évite d'avoir à traiter les segments isolés dans l'image et de recombinaison ces segments pour former un symbole complet.



Figure 2.3 Robot Pioneer 2 en face d'un symbole

En tenant compte de l'ensemble des lettres (majuscules et minuscules), des chiffres et des signes possibles, un nombre considérable de symboles peuvent être utilisés. Il fut donc décidé de restreindre cet ensemble à des symboles pouvant servir aux expérimentations diverses du laboratoire, soit les premières lettres des noms du robot (A, C, V, H, J, L), les quatre points cardinaux (en anglais) (N, S, E, W), les chiffres de 0 à 9, les flèches dans quatre directions (\uparrow , \downarrow , \leftarrow , \rightarrow) et un symbole référant à une station de recharge (\swarrow). Un total de 25 symboles est donc utilisé, ce qui constitue un ensemble suffisamment grand pour la validation des objectifs du projet sans alourdir inutilement les expérimentations.

3 EXTRACTION DU SYMBOLE

La première étape requise pour arriver à faire lire des symboles par un robot consiste à lui permettre d'extraire leur présence dans les images. La section 3.1 décrit les indices visuels pouvant servir à réaliser cette extraction et présente le choix effectué pour les travaux du mémoire, c'est-à-dire la couleur. Ensuite, la section 3.2 décrit les différents formats de représentation des couleurs. La section 3.3 expose l'algorithme conçu pour la perception des couleurs dans l'image, suivie par la section 3.4 de l'algorithme de segmentation utilisé pour l'extraction des symboles. Enfin, la section 3.5 présente les performances observées pour cette première étape de l'approche visant à l'interprétation visuelle de symboles par un robot mobile.

3.1 Indices visuels pour l'extraction d'un symbole dans une image

Plusieurs indices visuels existent pour percevoir des éléments dans une image. La couleur, les contours et les textures en sont des exemples [BRUCE 2000]. Les prochaines sections décrivent ces indices et exposent leurs contraintes respectives.

3.1.1 Perception de la couleur

La perception des couleurs est la technique la plus utilisée lorsqu'il s'agit de traitement en temps réel des images par des robots mobiles. Elle permet d'identifier des objets qui se trouvent dans l'environnement à partir de leur couleurs (déterminées à l'avance), sans se soucier véritablement de la forme des objets. Parmi les différentes approches, les plus populaires catégorisent les couleurs de chaque pixel de l'image en différentes classes ou groupes selon des seuils pré-établis [BRUCE 2000].

À cause de sa simplicité, l'utilisation de seuils constants (formant une boîte rectangulaire dans un espace de couleur en trois dimensions) est très fréquent. La figure 3.1 illustre cette partition pour un espace de couleur de format RGB (*Red, Green, Blue*). Un pixel est classifié en fonction de son appartenance à cette région délimitée par des seuils dans l'espace RGB.

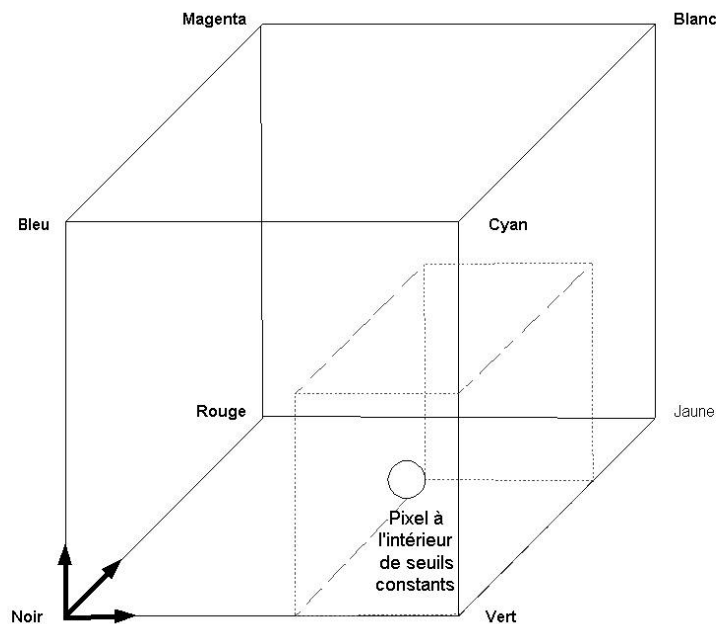


Figure 3.1 Espace de couleur RGB avec une région définie par seuils constants

Une autre approche consiste à utiliser l'algorithme des K-plus-proches-voisins (*K-Nearest Neighbor*) pour la classification des couleurs. Dans ce cas, plusieurs pixels pré-classifiés sont employés, chacun ayant une position unique dans l'espace de couleur. Pour obtenir la classification d'un nouveau pixel, la liste des K-plus-proches-voisins est établie et la couleur du pixel est trouvée à partir de la combinaison linéaire des couleurs des K voisins.

Finalement, des techniques statistiques sont aussi utilisées. Il s'agit de conserver en mémoire une table contenant les probabilités, pour chaque valeur de pixel, d'appartenir à chacune des couleurs à percevoir. Ceci permet d'obtenir une forme arbitraire au niveau des

seuils d'appartenance des couleurs. Par contre, la limitation de cette technique est que les tables qui contiennent les probabilités nécessitent beaucoup de mémoire.

3.1.2 Détection de contours

La détection des contours utilise les contrastes d'intensité de couleurs d'un pixel avec ses voisins pour déterminer les traits importants de l'image. Cette technique est très utilisée en vision artificielle et en robotique, mais demande beaucoup de calculs et nécessite du matériel spécialisé pour obtenir un temps de traitement qui respecte les contraintes temps réel des robots. En grande partie, l'importante quantité de calculs est causée par le traitement de chaque pixel avec ses huit voisins, ce qui nécessite au moins huit fois plus de calculs que le traitement des pixels individuels (comme pour la couleur par exemple).

De plus, la complexité des algorithmes d'extraction des contours réside dans le fait que la notion de contour est vague. Les méthodes les plus populaires utilisent généralement des filtres passe-haut à phase linéaire [MITRA 2000]. Ces filtres sont habituellement l'extension de filtres à une dimension, qui sont mieux connus et appliqués par la convolution du filtre en deux dimensions avec l'image. Les opérateurs de filtrage les plus utilisés se nomment les opérateurs de Sobel [BOVIK 2000] et sont représentés par des matrices de dimensions impaires (3×3 , 5×5 , etc.). Les dimensions impaires sont nécessaires pour que le pixel traité soit centré dans la matrice. L'application des opérateurs de Sobel, qui consiste à faire la somme des différences pondérées des pixels voisins avec le pixel central, résulte en un signal qui représente les contours. L'application de ces filtres donne une image avec des contours plus ou moins bien définis. Ces niveaux représentent un contour montant ou descendant. La plupart du temps, il s'agit de spécifier des seuils pour identifier les contours plus prononcés qu'il faut traiter. Par contre, la difficulté est que bien

souvent, les contours dans l'image ne sont pas clairement définis et l'algorithme ne réussit pas à les détecter.

3.1.3 Perception des textures

Les textures sont en fait des patrons qui se répètent dans l'image et qui doivent être discriminés. Avec les textures, il est possible de déterminer avec précision la composition des objets et leur position dans l'image. Cependant, la méthode est encore au stade de la recherche fondamentale et nécessite encore plus de calculs que l'extraction des contours. Elle n'est donc pas appropriée pour notre approche.

3.1.4 Choix de la méthode pour l'extraction du symbole

Bien qu'une approche combinant les couleurs, les contours et les textures présenterait une approche robuste pour l'extraction d'un symbole dans une image, la contrainte de traiter l'information rapidement avec un ordinateur embarqué limite nos choix. Ainsi, seule la technique de perception des couleurs est utilisée par notre approche.

3.2 Représentation des couleurs

Les couleurs peuvent être représentées sous différents formats. Dans tous les cas, chaque format présente des avantages et des inconvénients. Les trois formats étudiés dans cette section sont le RGB, le YUV et le HSV.

- **RGB (*Red, Green, Blue*)**. Le format de couleur RGB consiste en trois couleurs primaires : le rouge, le vert et le bleu. Les composantes spectrales de chacune de ces couleurs de base s'additionnent pour former une couleur résultante. Le format RGB est représenté par un cube à trois dimensions avec le rouge, le vert et le bleu à chaque coin du cube, comme montré à la figure 3.1. Le noir est à l'origine du cube (0, 0, 0) et le

blanc est à l'opposé du cube et contient les valeurs maximales de chaque composant RGB. Une couleur à percevoir est donc identifiée par des seuils sur chacun des composants. Les seuils constants définissent une boîte rectangulaire dans l'espace de couleur.

La gradation des couleurs sur les composants R, G et B dépend du nombre de bits alloués pour chaque composant. Dans le cas d'une image RGB à 24 bits, 8 bits sont alloués par composant, ce qui résulte en 256 niveaux (2^8) pour chaque couleur primaire. Ainsi, les coordonnées de la couleur blanche pour une image RGB à 24 bits sont (255, 255, 255). L'échelle noir et blanc est en fait la ligne diagonale qui traverse le cube du noir vers le blanc sur la figure 3.1.

Le format RGB est très utilisé dans les cartes graphiques pour les ordinateurs. Cependant, il n'est pas nécessairement le plus adéquat dans toutes les applications pour le traitement d'images. En effet, étant donné que les composants RGB sont corrélés, il est difficile d'obtenir une bonne séparation des couleurs à partir d'algorithmes utilisant des seuils constants.

- **YUV (*Luminance, Chrominance Bleu, Chrominance Rouge*)**. Le format YUV sépare l'information d'intensité des couleurs (Y) de l'information des couleurs (U et V). C'est le format qui est utilisé pour la transmission des images pour la télévision. De cette manière, les téléviseurs qui fonctionnent en noir et blanc n'ont qu'à utiliser le composant Y, tandis que les téléviseurs couleurs doivent aussi incorporer l'information U et V pour former l'image en couleur. Il n'est pas vraiment intéressant de représenter l'espace YUV sous forme de cube comme l'espace RGB puisque les arrêtes ne

représentent pas les couleurs de base. La transformation de YUV à RGB est une transformation linéaire décrite par les équations 3.1 et 3.2.

$$\begin{array}{l} \text{RGB} \rightarrow \text{YUV} \\ \text{(normalisé)} \end{array} \quad \begin{bmatrix} Y \\ U \\ V \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ -0.16874 & -0.33126 & 0.5 \\ 0.5 & -0.41869 & -0.8131 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} \quad (3.1)$$

$$\begin{array}{l} \text{YUV} \rightarrow \text{RGB} \\ \text{(normalisé)} \end{array} \quad \begin{bmatrix} R \\ G \\ B \end{bmatrix} = \begin{bmatrix} 1.00000 & 0 & 1.402 \\ 1.00000 & -0.34414 & -0.71414 \\ 1.00000 & 1.772 & 0 \end{bmatrix} \begin{bmatrix} Y \\ U \\ V \end{bmatrix} \quad (3.2)$$

- **HSV (*Hue*, *Saturation*, *Value*).** Le format HSV a été créé pour être plus facile à utiliser par les infographistes. Le composant H (*Hue*) est une mesure d'angle similaire à la palette de couleur des logiciels de dessin. La valeur d'angle 0° réfère à la couleur rouge, 120° au vert et 240° au bleu. Les plans horizontaux sont des hexagones avec à ses arrêtes les couleurs primaires et secondaires (rouge, jaune, vert, cyan, bleu, magenta). Le composant S (*Saturation*) indique l'intensité de la couleur de 0 (sans couleur, noir et blanc) au maximum d'intensité pour la couleur. Finalement, le composant V (*Value*) représente la brillance (*brightness*), de 0 au maximum d'éclat pour la couleur. La représentation graphique de l'espace HSV est illustrée à la figure 3.2. Les transformations de HSV à RGB sont des transformations non-linéaires décrites par les équations 3.3 et 3.4.

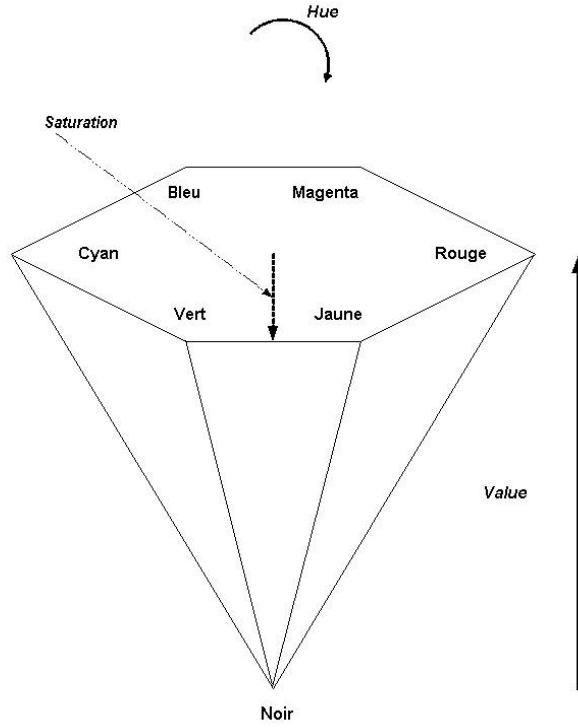


Figure 3.2 Espace de couleur HSV

RGB→HSV
(normalisé)

$$\begin{aligned}
 r &= \frac{R}{R+G+B} \\
 g &= \frac{G}{R+G+B} \\
 b &= \frac{B}{R+G+B} \\
 H &= \cos^{-1} \left\{ \frac{\frac{1}{2}[(r-g) + (r-b)]}{\left[\frac{1}{4}[(r-g)^2 + (r-b)(g-b)] \right]^{\frac{1}{2}}} \right\} \\
 S &= 1 - \frac{3}{r+g+b} [\min(r, g, b)] \\
 V &= \frac{1}{3}(r+g+b)
 \end{aligned} \tag{3.3}$$

$$\begin{aligned}
& r = \frac{1}{3}(1 - S) \\
\text{HSV} \rightarrow \text{RGB} & \quad g = \frac{1}{3} \left[1 + \frac{S \cos(H)}{\cos(60 - H)} \right] \\
\text{(normalisé)} & \quad b = 1 - (r + g)
\end{aligned} \tag{3.4}$$

Les formats disponibles pour la carte de capture PXC200 du robot sont le RGB et le YUV. Effectuer les transformations de ces formats au format HSV demanderait beaucoup de calculs sans apporter des avantages intéressants. De plus, le format YUV est disponible en format YUV422 seulement. Cela signifie que les composants U et V sont transmis seulement à tous les deux pixels, contrairement à Y qui est transmis à tous les pixels. Il faut alors restaurer les valeurs manquantes de U et de V en faisant de l'interpolation avec les valeurs voisines de U et V pour obtenir tous les composants d'un pixel. Par contre, la carte de capture est en mesure de fournir les composants RGB pour chacun des pixels, sur 15, 16 ou 24 bits. Dans le but de minimiser le temps de traitement et l'utilisation de l'espace mémoire, le format RGB15 (5 bits par couleur) a été choisi pour satisfaire les contraintes de notre projet. Le format RGB15 donne la possibilité de 2^{15} couleurs différentes, ce qui est suffisant pour notre projet puisqu'il n'est pas nécessaire d'avoir une grande précision pour les différentes teintes.

3.3 Algorithme de perception des couleurs pour les symboles

Comme mentionné précédemment, le principal problème avec la perception des couleurs avec le format RGB réside dans le fait que les algorithmes utilisent des seuils constants. Puisque l'étalement d'une couleur dans l'espace RGB est difficile à contenir à l'intérieur d'une région de forme rectangulaire, prisme ou cubique, il faut avoir recours à d'autres techniques.

La technique développée est une adaptation de l'algorithme décrit dans [BRUCE et coll. 2000]. L'algorithme de Bruce stocke les seuils constants de format YUV dans trois tables en mémoire (*lookup tables*). Puisque chaque composant est défini sur 8 bits, chacune des tables a 256 entrées. L'inclusion d'une couleur consiste à inscrire un (1) dans la table pour les entrées faisant partie des seuils pour celle-ci. L'identification de la couleur d'un pixel consiste à vérifier, pour chacun des composants YUV du pixel, si un (1) se trouve dans les tables. L'opération consiste donc à effectuer trois lectures dans les tables, et d'effectuer deux ET logiques pour obtenir l'appartenance (1) ou non (0) d'un pixel à une couleur. De cette façon, il n'est pas nécessaire d'évaluer de multiples conditions définies par des seuils pour chaque couleur à percevoir, telles que :

$$\begin{aligned}
 & \text{SI } ((Y > \text{MIN_Y ET } Y < \text{MAX_Y}) \\
 & \quad \text{ET } (U > \text{MIN_U ET } U < \text{MAX_U}) \\
 & \quad \text{ET } (V > \text{MIN_V ET } V < \text{MAX_V})) \\
 & \text{ALORS COULEUR} = \text{COULEUR_1.}
 \end{aligned} \tag{3.5}$$

Les branchements (SI-ALORS) sont ainsi remplacés par la consultation directe des tables et par deux opérateurs logiques, ce qui est beaucoup plus rapide à exécuter par le processeur. De plus, l'algorithme utilise un entier non-signé de 32 bits pour stocker les seuils de perception des couleurs permettant ainsi de représenter de façon compacte les seuils pour 32 canaux de couleur. De cette façon, les trois tables représentent l'appartenance ou non à une couleur (à l'intérieur de seuils constants) pour toutes les valeurs de YUV.

Au lieu d'utiliser trois tables de 256 valeurs indexées par les composants YUV, notre algorithme emmagasine l'appartenance ou non à une couleur dans une seule table indexée par les composants RGB. Utiliser une valeur RGB à 15 bits nécessite une table de 2^{15} éléments (32768 éléments). Avec chacun des composants représentés sur 5 bits, l'index à la table est alors de forme 0RRRRRGGGGGBBBBB (en binaire). Cette valeur est en fait

celle qui est directement fournie par la carte de capture PXC200. La figure 3.3 illustre le fonctionnement de la table. Pour chaque index dans celle-ci, il est possible de stocker de l'information pour 32 canaux de perception de couleur simultanément, canaux identifiés par la position du bit dans l'entier non signé de 32 bits comme pour l'algorithme de Bruce. Il est important de noter qu'un même index RGB15 peut être associé à plusieurs canaux. Par contre, au lieu de nécessiter trois lectures dans des tables et deux opérations logiques, notre algorithme requiert une seule recherche dans la table. Elle demande toutefois plus d'espace mémoire (128 Ko par rapport à 3 Ko pour l'algorithme de Bruce). Pour notre projet, la rapidité d'exécution est l'objectif souhaité et l'espace-mémoire additionnel requis n'entraîne aucune complication particulière.

Le numéro du bit correspond au numéro de canal de perception d'une couleur. Capacité de stocker 32 canaux (32 bits).

Index de la table (RGB15)	0	1	2	3	...	29	30	31
RGB15 VALEUR = 0	1	0	0	0	...	1	0	0
RGB15 VALEUR = 1	0	1	0	0	...	0	0	0
RGB15 VALEUR =
RGB15 VALEUR = 32766	0	1	0	0	...	0	0	0
RGB15 VALEUR = 32767	0	0	1	0	...	0	0	0

Figure 3.3 Table d'appartenance des couleurs

Il reste maintenant à expliquer comment les couleurs à percevoir sont initialisées dans la table. Un processus à deux étapes est réalisé :

- 1) **Conversion de seuils établis dans le format HSV au format RGB.** La plupart des logiciels de traitement d'image (*Corel Draw*, *Gimp*, *Photoshop*, etc.) permettent de sélectionner des couleurs dans l'espace de couleur HSV. Puisque la représentation HSV rend facile la séparation des couleurs pour l'humain, des seuils préliminaires sont choisis dans le format HSV pour ensuite être transformés en format RGB. En ayant les seuils dans le format RGB par la transformation HSV→RGB (selon les équations 3.3 et 3.4), il est possible d'initialiser la table d'appartenance des couleurs facilement. Cette étape ne s'effectue qu'une seule fois lors du choix des couleurs pour les canaux. Ainsi, les avantages de la représentation HSV sont utilisés pour déterminer des seuils RGB qui seraient difficiles à déterminer visuellement.
- 2) **Ajustement des couleurs en les sélectionnant directement à partir d'une image.** Cette étape s'effectue avec l'aide d'une interface-usager montrée à la figure 3.4. L'utilisateur peut ajouter ou supprimer certaines couleurs dans la table d'appartenance directement à partir d'une image. Pour sélectionner ou enlever les couleurs dans la table, l'utilisateur sélectionne avec la souris une région de forme rectangulaire dans l'image. Toutes les valeurs de pixels (RGB) à l'intérieur du rectangle choisi sont utilisées pour mettre à jour l'appartenance au canal de couleur dans la table. Il s'agit tout simplement de mettre un (1) ou un (0) au bon index RGB à la position du canal que l'utilisateur est en train de modifier. L'interface affiche alors le résultat de l'entraînement sur l'image pour bien identifier les pixels qui font partie du canal de couleur. Cette interface s'est avérée très utile et efficace pour l'entraînement des

couleurs sous différentes conditions d'illumination. Plusieurs images étaient prises sous différents angles par rapport à la source de lumière, afin de bien déterminer toutes les valeurs RGB possibles pour la couleur à percevoir.

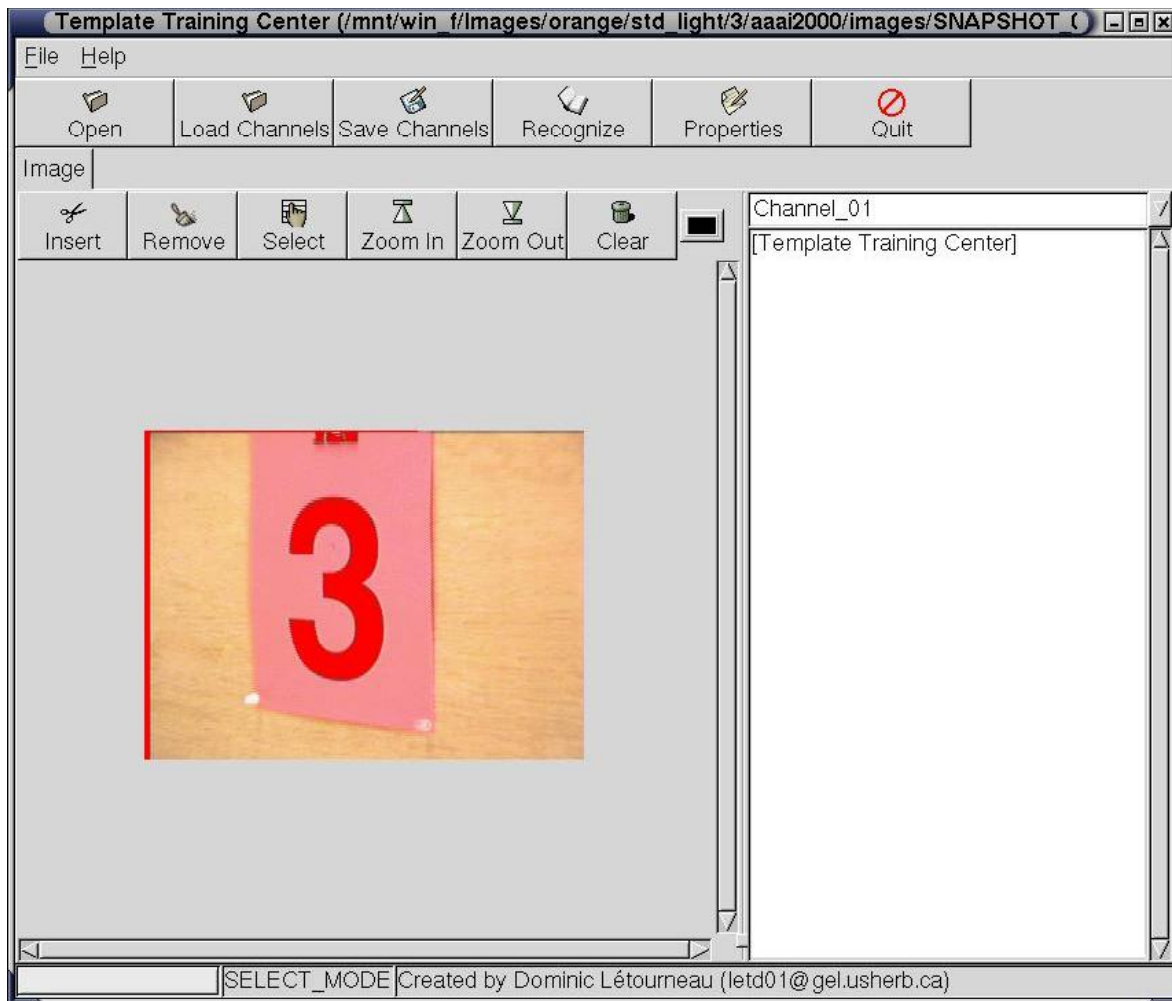


Figure 3.4 Interface-usager pour l'ajustement des couleurs (par exemple le noir)

Enfin, la figure 3.5 représente, dans l'espace RGB, l'appartenance pour les quatre couleurs utilisées dans nos expérimentations : le noir, le bleu, le rose et l'orange. Cette figure illustre clairement que la forme des régions pour chaque canal de couleur peut difficilement être représentée par des seuils constants. En fait, il est possible de constater que les seuils HSV transformés en seuils RGB forment une région bien délimitée. Pour le noir par exemple,

ceci se traduit par une région hexagonale autour de (0, 0, 0). L'interface-usager a permis d'ajouter les valeurs qui forment un «V» au dessus de l'hexagone, pour une détection plus robuste aux variations des conditions d'illumination. Ainsi, la méthode utilisée arrive à former des zones de perception de couleur de formes quelconques et qui nécessiteraient plusieurs petites régions à seuils constants pour bien fonctionner.

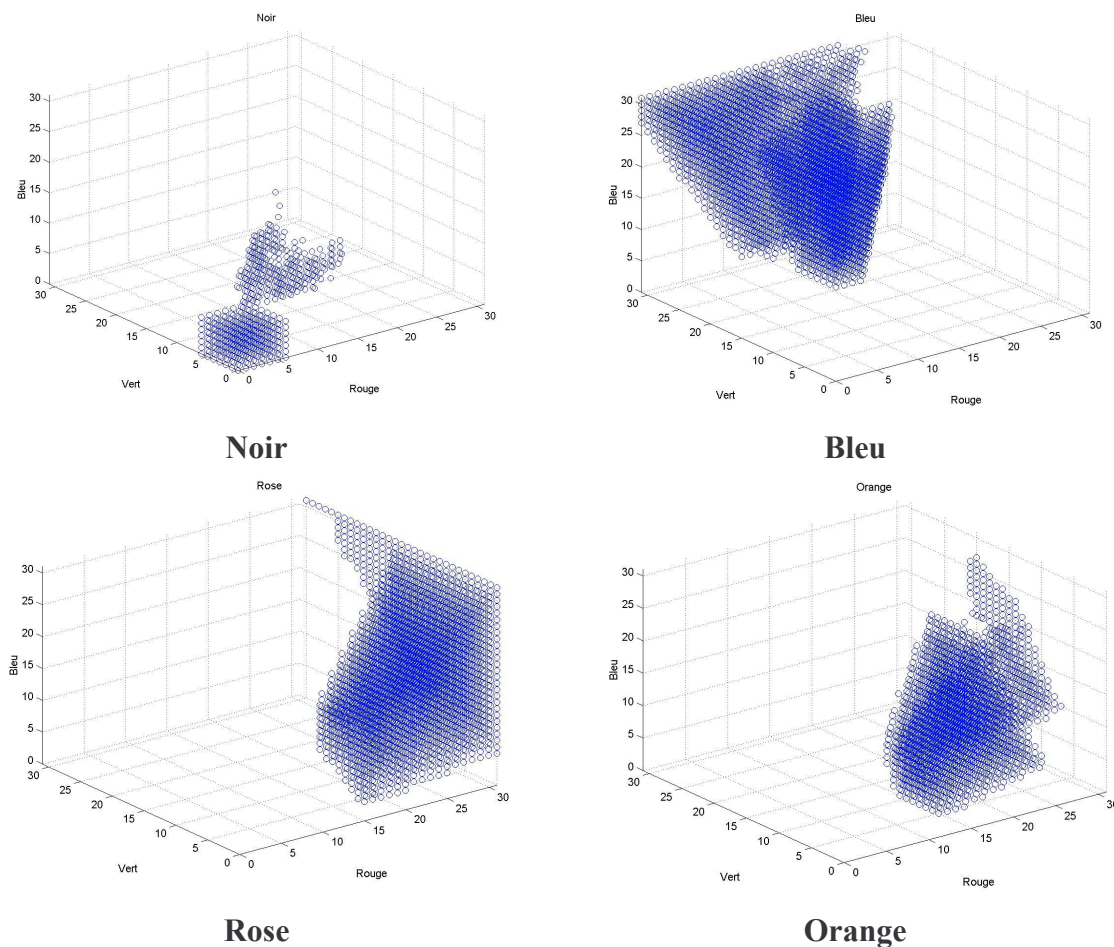


Figure 3.5 Représentation dans l'espace RGB des quatre couleurs utilisées

3.4 Algorithme de segmentation des couleurs

La segmentation des couleurs consiste à regrouper les pixels de même couleur qui sont voisins dans l'image. Cette technique permet de traiter les composants de l'image en région

au lieu de traiter chaque pixel individuellement. Notre algorithme de segmentation des couleurs fonctionne en deux étapes :

- 1) L'image est d'abord divisée en cellules, avec chaque cellule regroupant les pixels adjacents appartenant à la même couleur sur une même ligne dans l'image. Ainsi, au lieu d'avoir plusieurs pixels semblables qui se suivent, les pixels sont groupés en une seule cellule. Les coordonnées de départ ($x1, y1$) et les coordonnées de la fin de la cellule ($x2, y2=y1$) sont déterminées. Pour chaque ligne de l'image, une liste de cellules par canal de perception de couleur est mémorisée pour un total de 7680 listes (32 canaux de couleur pour chacune des 240 lignes).
- 2) Lorsque tous les pixels sont regroupés en cellules, il faut lier les cellules qui sont connectées d'une ligne à l'autre pour ainsi former des régions rectangulaires sur l'image. Les cellules qui sont connectées sont celles qui sont sur les lignes immédiatement au-dessus ou en-dessous, et dont la distance de séparation horizontale maximale ne dépasse pas un pixel de large. Pour chaque canal de couleur, l'algorithme fonctionne de manière récursive et débute à la première ligne de l'image. Pour chaque cellule traitée, l'algorithme tente de connecter les cellules de la ligne précédente, pour ensuite tenter de connecter les cellules de la ligne suivante. Cette façon de faire permet de reconnecter les cellules d'une même ligne qui sont liées par des cellules au-dessus ou en-dessous d'elles. Les cellules connectées sont enlevées des listes de cellules de l'image et sont incluses à l'intérieur d'une région rectangulaire en mettant à jour les coordonnées du coin supérieur gauche ($x1, y1$), du coin inférieur droit ($x2, y2$), le centre de gravité et l'aire de cette région. La figure 3.6 illustre le processus en indiquant l'ordre de visite des cellules par le chiffre à gauche des cellules. Les pointillés signifient que l'algorithme a groupé ces cellules pour former une région rectangulaire. Lorsqu'il

n'y a plus de cellules à connecter pour une région, un autre rectangle pour représenter une autre région est créé et l'algorithme redémarre sa recherche pour connecter d'autres cellules récursivement. La recherche de régions s'effectue jusqu'à ce que toutes les cellules de l'image aient été traitées pour chaque canal de couleur. Toutes les régions rectangulaires sont conservées en mémoire pour un traitement ultérieur et sont triées par aire (du regroupement ayant le plus grand nombre de pixels à celui regroupant le plus petit nombre de pixels).

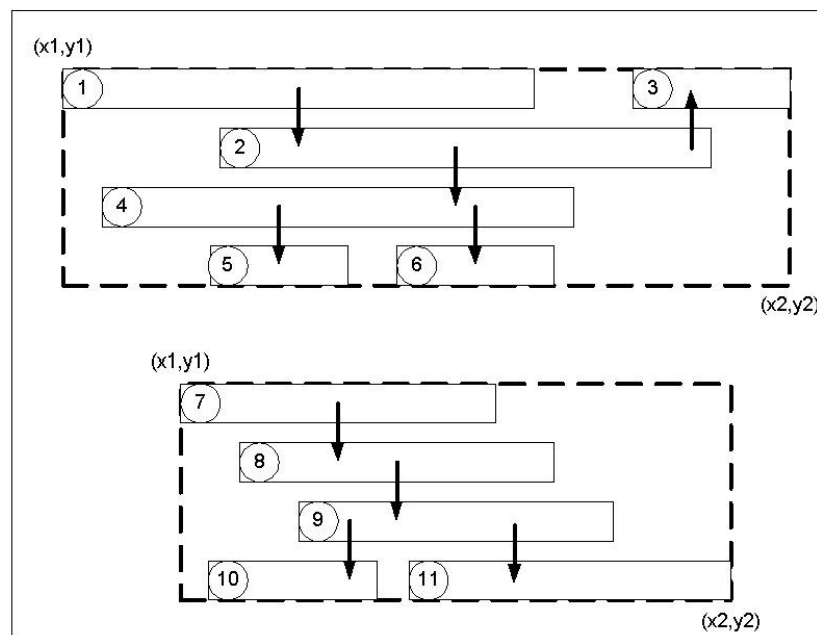


Figure 3.6 Algorithme de connexion des cellules de même couleur

3.5 Analyse des performances pour l'extraction des couleurs

Les techniques décrites précédemment pour la représentation des couleurs et la segmentation permettent d'obtenir un temps de traitement adéquat des images à traiter. L'objectif est d'arriver à traiter un nombre suffisant d'images par seconde pour que le robot puisse agir suffisamment rapidement selon sa dynamique et celle de l'environnement. Le traitement des images est réalisé de façon concurrente (en parallèle) avec les algorithmes de

contrôle du robot. Il faut donc penser à minimiser la charge de calculs afin de laisser du temps de microprocesseur pour la prise de décision par le robot. En mode requête pour la saisie des images, 10 images par seconde sont traitées en utilisant 20% du temps processeur. Ceci est suffisant pour le robot, car son temps de réaction (incluant la prise de l'image et l'écriture des commandes) est de 100 msec (selon les indications fournies à la section 2.1). Des tests sur un autre système informatique similaire à celui du robot *Pioneer 2*, mais capable de capturer des images en mode continu (30 images par seconde), révèlent que l'algorithme y arrive en utilisant 50% du temps processeur. L'approche peut donc servir à traiter des images à un rythme plus élevé.

4 POSITIONNEMENT DU ROBOT ET DE LA CAMÉRA

En étant capable de percevoir les couleurs, le robot est en mesure de détecter la présence potentielle d'un symbole. Il doit par la suite se positionner adéquatement en face du symbole et en obtenir une image claire pour l'étape de reconnaissance. En fait, puisque le robot est mobile et peut percevoir un symbole sous différents angles et à différentes distances, il est nécessaire de bien positionner le robot ainsi que sa caméra afin de minimiser la distorsion du symbole [NOTAKE et coll. 2000] et maximiser les chances d'une bonne reconnaissance.

L'approche utilisée pour le positionnement du robot et de sa caméra s'effectue en deux étapes. La première, expliquée à la section 4.1, met en œuvre un mécanisme de contrôle des déplacements du robot. La deuxième étape, décrite à la section 4.2, consiste à contrôler les déplacements horizontaux, verticaux et d'agrandissement de la caméra pour centrer le symbole dans l'image.

4.1 Contrôle des déplacements du robot

L'approche retenue pour le contrôle du robot s'inspire de l'approche comportementale [ARKIN 1998] à laquelle s'ajoute un module de traitement de plus haut niveau pour la reconnaissance du symbole (décrit au chapitre 5). Les modules comportementaux requis permettent au robot de se déplacer en évitant les obstacles, de percevoir des symboles et d'en capturer des images convenables. La figure 4.1 illustre l'architecture décisionnelle implémentée sur le robot.

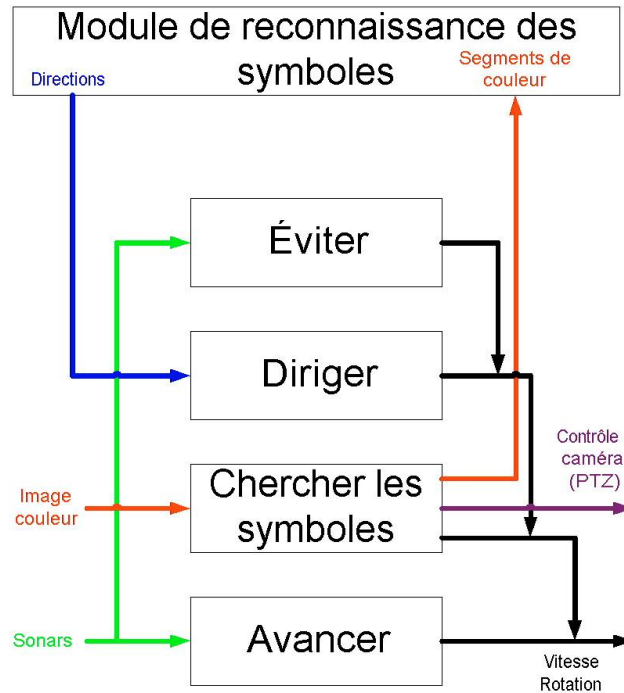


Figure 4.1 Organisation des modules décisionnels du robot

Les modules *Avancer* et *Éviter* permettent au robot de se déplacer en évitant les obstacles à partir des lectures de sonars. Le module *Chercher les symboles* examine la présence de couleurs propres aux symboles à partir des images obtenues par la caméra selon l'approche décrite au chapitre 3. Lorsqu'une région de couleur noire est complètement entourée par une région avec une couleur de fond (bleu, rose ou orange), ce module comportemental stoppe le robot. Par la suite, il tente de positionner le centre de gravité du symbole au milieu de l'image en donnant des commandes de déplacements horizontaux et verticaux à la caméra et en agrandissant l'image avec le zoom. Le module *Chercher les symboles* peut également envoyer des commandes de rotation au robot dans le but que le symbole potentiel soit en face du robot et que sa caméra soit perpendiculaire au symbole. Dans ce cas, c'est le robot qui « tourne » au lieu de la caméra. La procédure détaillée est décrite à la section 4.2. Une fois le symbole traité, le module comportemental *Chercher les symboles*

permet au robot de reprendre ses déplacements et repositionne la caméra vers l'avant, sans zoom.

Enfin, le module comportemental *Diriger* permet de spécifier des commandes (par exemple rotation $+90^\circ$, avancer d'un mètre, etc.) selon les traitements réalisés par un module de plus haut niveau. Pour notre application, *Diriger* reçoit des commandes de rotation et de déplacement pour positionner le robot par rapport au symbole et pour exécuter l'action qui découle du symbole reconnu.

L'arbitration des modules comportementaux s'effectue par *Subsumption* [BROOKS 1986]. Sur la figure 4.1, ceci signifie que la priorité des comportements augmente du bas vers le haut, et que les commandes envoyées par les modules prioritaires remplacent celles des modules comportementaux au bas de la hiérarchie. Ainsi, le robot donne une priorité absolue à l'évitement d'obstacles, qu'il soit en mouvement ou non.

4.2 Positionnement de la caméra

Une fois le robot immobilisé, l'objectif est d'arriver à obtenir une image adéquate (pour la reconnaissance) du symbole à identifier. La figure 4.2 montre en a) l'image initiale d'un symbole perçu lors de l'immobilisation du robot, et en b) l'image du symbole avec la meilleure résolution possible en contrôlant les paramètres Pan-Tilt-Zoom de la caméra.

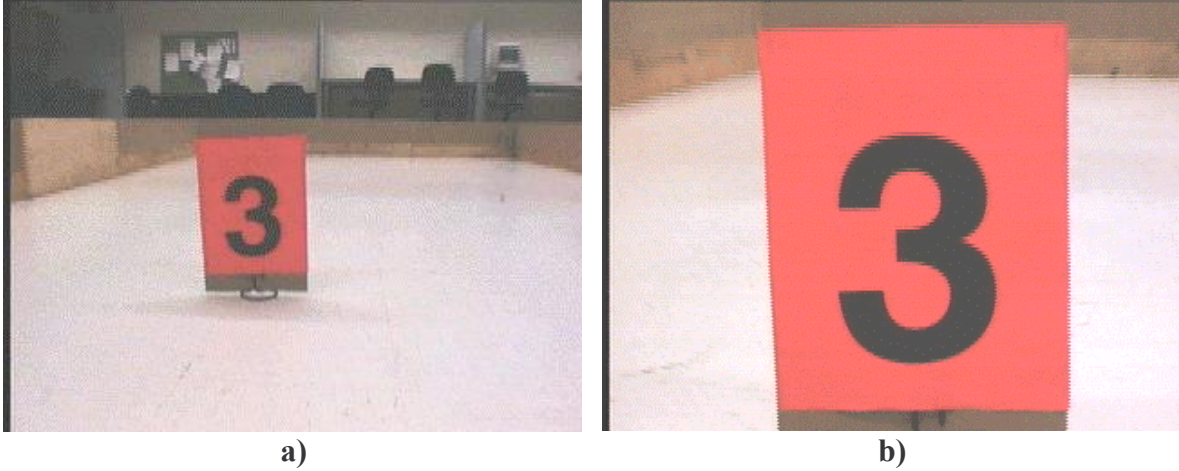


Figure 4.2 Images a) avant et b) après le positionnement de la caméra

La figure 4.3 illustre l'algorithme utilisé pour le positionnement du symbole dans l'image avec la caméra. Cet algorithme est évalué à chaque cycle d'exécution du robot, et retourne des commandes de Pan, de Tilt et de Zoom à toutes les 100 msec.

La première étape consiste à extraire la position du centre de gravité du symbole dans l'image sur laquelle fut réalisée l'extraction des couleurs. L'image traitée représente par un (1) les pixels faisant partie du symbole et par un (0) les autres. Si une région de couleur noire est entourée d'une région de couleur de fond admissible pour les symboles (orange, bleu ou rose), un symbole potentiel est présent dans l'image. Les coordonnées (x_c, y_c) (colonne et ligne dans l'image) de la région en noir sont alors calculées par les équations 4.1 et 4.2.

$$x_c = \frac{\sum_{x=x1}^{x2} \sum_{y=y1}^{y2} xR(x, y)}{\sum_{x=x1}^{x2} \sum_{y=y1}^{y2} R(x, y)} \quad (4.1)$$

$$y_c = \frac{\sum_{x=x1}^{x2} \sum_{y=y1}^{y2} yR(x, y)}{\sum_{x=x1}^{x2} \sum_{y=y1}^{y2} R(x, y)} \quad (4.2)$$

Les coordonnées (0,0) se situent au coin supérieur gauche de l'image et croissent de gauche à droite et de haut en bas. Si plusieurs symboles sont présents dans l'image, seul le plus grand symbole (celui avec l'aire la plus grande) est considéré. Obtenir le symbole le plus grand dans l'image est facile puisqu'à la suite de l'extraction des couleurs, les symboles sont triés par région selon l'aire, comme mentionné au chapitre 3.

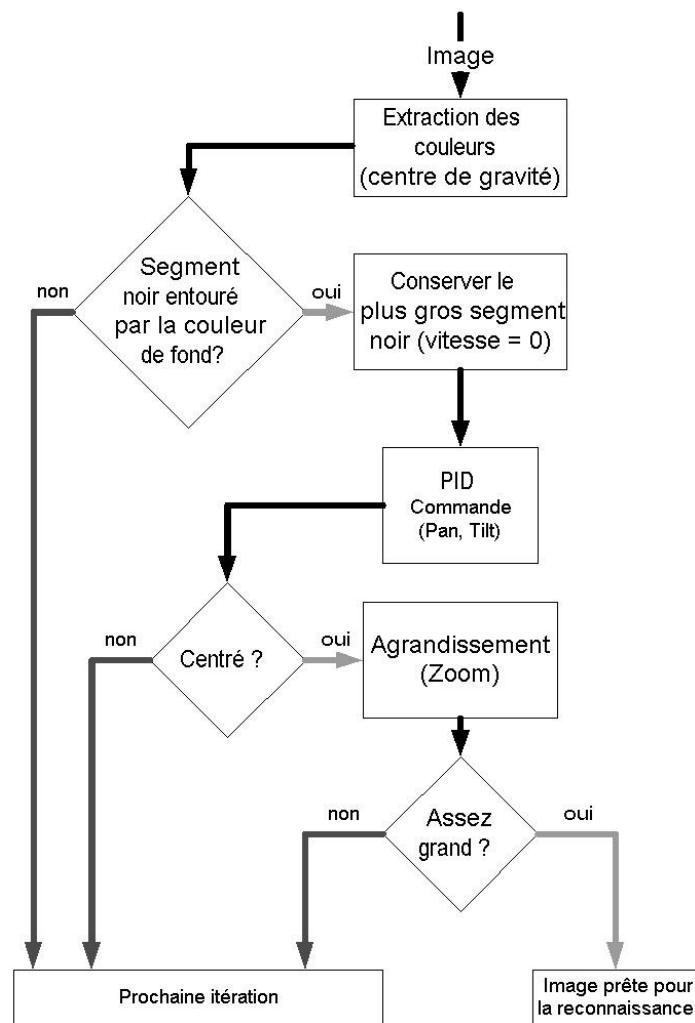


Figure 4.3 Algorithme pour centrer le symbole

À partir des coordonnées (x_c, y_c) du centre de gravité, la seconde étape consiste à modifier l'orientation horizontale (Pan) et verticale (Tilt) de la caméra afin de centrer le symbole dans l'image. Le déplacement en pixels à réaliser est donc $\Delta x = 320/2 - x_c$, et $\Delta y = 240/2 - y_c$. Pour effectuer ces déplacements, un contrôleur PID (Proportionnel-Intégral-Dérivé) est utilisé. La difficulté ici est que les commandes de déplacements horizontaux et verticaux de la caméra sont fonction du zoom utilisé pour l'image traitée. En effet, plus le zoom est élevé, moins les déplacements doivent être grands afin de ne pas perdre le symbole de vue. De plus, cette variation n'est pas linéaire sur toute la plage d'opération du zoom. La figure 4.4 illustre cette observation. Ce graphique fut obtenu en mesurant la longueur en pixels d'un objet placé à une distance fixe à l'avant de la caméra et en variant les commandes de zoom. Les commandes qui sont envoyées à la caméra varient de 0 à 1024. La figure 4.4 illustre la relation entre les commandes envoyées à la caméra et le facteur agrandissement (A) réel, qui est obtenu en divisant la largeur en pixels de l'objet à un zoom donné par la largeur en pixels de l'objet à un zoom nul. Cette courbe fut ensuite modélisée mathématiquement par la méthode des moindres carrés. Le résultat de la modélisation est présenté à l'équation 4.3. Le facteur d'agrandissement obtenu permet de considérer l'effet sur Δx et Δy (les distances en pixels du symbole par rapport au centre de l'image) pour calculer les déplacements à réaliser, tels que décrits par les équations 4.4 et 4.5. La modélisation avec l'ordre 3 (équation 4.3) de l'agrandissement par rapport à la commande de zoom nous permet de tracer une courbe qui reflète exactement les données prises par expérimentation. Un ordre inférieur ne nous aurait pas permis d'obtenir une aussi bonne représentation.

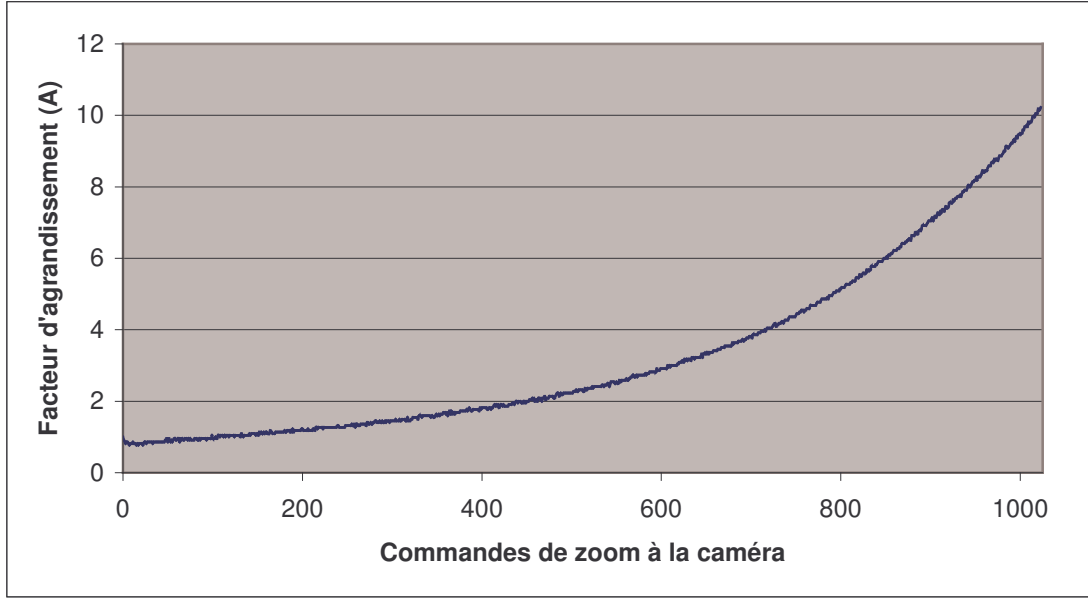


Figure 4.4 Facteur d'agrandissement de la caméra Sony EVI-D30

$$A = 0.68 + 0.0041zoom + 8.94 * 10^{-6} zoom^2 + 1.36 * 10^{-8} zoom^3 \quad (4.3)$$

$$\Delta x' = \frac{\Delta x_{image}}{A} \quad (4.4)$$

$$\Delta y' = \frac{\Delta y_{image}}{A} \quad (4.5)$$

Les équations 4.6 et 4.7 présentent les coefficients du contrôleur PID en Pan et en Tilt. Une autre difficulté ici est que le système de vision du robot ne permet pas de retourner la position exacte de la caméra. L'algorithme doit donc envoyer des commandes qui seront en mesure d'être complétées après chaque cycle de traitement. Les coefficients ont donc été ajustés par essais et erreurs, en plaçant un symbole à l'avant du robot pour différentes valeurs de zoom. Bien qu'il nous est impossible de démontrer si les contrôleurs sont optimaux ou non, les essais confirment que la caméra arrive à bien se stabiliser avec un symbole au centre de l'image, générant de très légères oscillations. Un contrôleur avec seulement une constante proportionnelle aurait été suffisant. Cependant, les composants

dérivés et intégrales du PID permettent de suivre un sujet qui bouge et d'anticiper les mouvements de la caméra pour les prochaines itérations [OGATA 1997]. La partie dérivée du contrôleur permet de prendre de l'avance sur le mouvement du sujet à suivre. Elle accélère le temps de réponse et réduit les oscillations pour une entrée de type échelon. La partie intégrale du contrôleur augmente le gain à basse fréquence et améliore la précision de l'état stable du système, mais augmente le temps de réponse puisqu'elle réduit la bande passante. Ainsi, l'ajustement des coefficients nécessite de faire des compromis entre le temps de réponse et la stabilité du système. L'approche proposée permettrait donc à un robot de suivre des symboles qui sont en mouvement.

$$\Delta Pan = 3\Delta x' + 0.01 \int_t \Delta x' dt + 0.0075 \frac{\partial \Delta x'}{\partial t} \quad (4.6)$$

$$\Delta Tilt = 3\Delta y' + 0.01 \int_t \Delta y' dt + 0.0075 \frac{\partial \Delta y'}{\partial t} \quad (4.7)$$

Une fois le symbole centré dans l'image pour un zoom donné, la troisième étape consiste à augmenter le zoom de la caméra pour obtenir une image du symbole de résolution suffisante. Il est important de noter que les commandes de zoom peuvent être données simultanément aux commandes de Pan et de Tilt lorsque le symbole est centré dans l'image à partir de la boucle de traitement illustrée à la figure 4.3. Les commandes de zoom sont déterminées en fonction de la distance (d) du bord de l'image, selon les limites obtenues par les coordonnées $(x1, y1)$ du coin supérieur gauche et $(x2, y2)$ du coin inférieur droit de la région rectangulaire de l'image qui contient le symbole. La distance (d) est définie par la relation 4.8.

$$d = \min(\min(320 - x2, x1), \min(240 - y2, y1)) \quad (4.8)$$

L'heuristique utilisée pour le contrôle du zoom augmente la valeur du zoom lorsque le symbole est suffisamment centré et que la distance (d) est assez grande par rapport au bord de l'image. L'algorithme considère que le symbole est suffisamment centré lorsque la distance entre le centre de gravité du symbole et le centre de l'image est plus petit que 30 pixels. Dans tous les cas où la distance (d) entre les extrémités du symbole et le bord de l'image est faible (< 10 pixels), il faut diminuer la commande de zoom pour s'assurer que le symbole reste complètement visible dans l'image à la prochaine itération. Les commandes de zoom sont fixées de manière à éviter de trop grandes oscillations de Pan et de Tilt et varient elles aussi en fonction de l'agrandissement (A) du symbole dans l'image. Les incréments de zoom sont fixés empiriquement et jouent un rôle important pour la stabilité du système. Ils tiennent compte de la vitesse de réaction de la caméra. Les règles de décision suivantes sont utilisées :

$$\begin{aligned}
 &\text{SI } (|\Delta x| < 30 \text{ ET } |\Delta y| < 30) \text{ ALORS} \\
 &\quad \text{SI } (d > 30) \text{ ALORS} \\
 &\quad \quad \text{Zoom} = \text{Zoom} + 25 / A \\
 &\quad \text{FIN SI} \\
 &\text{FIN SI} \\
 &\text{SI } (d < 10) \text{ ALORS} \\
 &\quad \text{Zoom} = \text{Zoom} - 25 / A \\
 &\text{FIN SI}
 \end{aligned} \tag{4.9}$$

Finalement, l'image servant à la reconnaissance du symbole est prise lorsque le symbole dans l'image est centré et ne peut plus être agrandi sans être trop aux abords de l'image ($d < 10$).

En utilisant les algorithmes décrits précédemment, le tableau 4.1 présente les temps moyens requis pour le positionnement de la caméra en fonction de la distance et de la taille établie du symbole. La portée de perception maximale est de 10 pieds. Pour des distances plus

élevées, le symbole est alors trop petit dans l'image pour que l'algorithme puisse capter le symbole noir entouré de la couleur de fond.

De plus, les symboles qui sont positionnés à moins de 2 pieds sont non-perceptibles puisqu'ils ne se trouvent pas dans le champ de vision du robot. Il est important de noter que le temps de capture d'un symbole varie en fonction de la distance. Ceci s'explique par le fait que l'algorithme de positionnement doit donner plus de commandes pour centrer le symbole et l'agrandir jusqu'à sa résolution maximale. Le temps de positionnement moyen global est donc de 18.1 secondes, pour des images avec une résolution maximale du symbole.

TABLEAU 4.1 Temps de capture d'un symbole à différentes distances

Distance (pieds)	Temps (secondes)
2	8.4
3	9.3
4	12.1
5	15.0
6	17.8
7	21.1
8	24.4
9	27.1
10	27.6

5 RECONNAISSANCE DU SYMBOLE

L'utilisation des réseaux de neurones est courante dans le domaine de la reconnaissance des caractères. En effet, les réseaux à rétro-propagation à une couche cachée sont souvent utilisés pour la reconnaissance de caractères manuscrits ou en format d'imprimerie [HAYKIN 1999][BUNKE 1997]. Leur capacité de généralisation, leur simplicité et leur robustesse au bruit font de cette approche un excellent choix pour notre projet. La section 5.1 décrit les caractéristiques des réseaux de neurones utilisés dans nos expériences. La section 5.2 présente deux méthodes étudiées pour encoder les symboles perçus pour leur reconnaissance par réseau de neurones. Enfin, la section 5.3 présente les performances de reconnaissance pour différentes configurations de réseaux de neurones, pour chacune des deux méthodes d'encodage.

5.1 Réseaux de neurone à propagation avant avec apprentissage par rétro-propagation

Un réseau de neurones est composé de plusieurs unités (appelées neurones artificiels) qui sont connectées par des liens. Ces liens comportent des poids qui sont mis à jour par un algorithme d'apprentissage. Ces poids déterminent l'influence des unités entre elles afin de permettre au réseau de traiter adéquatement les données pour l'application en question.

Le modèle d'un neurone est donné à la figure 5.1 et illustre ses trois éléments importants : la fonction d'entrée, la fonction d'activation et la sortie (ou l'activation du neurone).

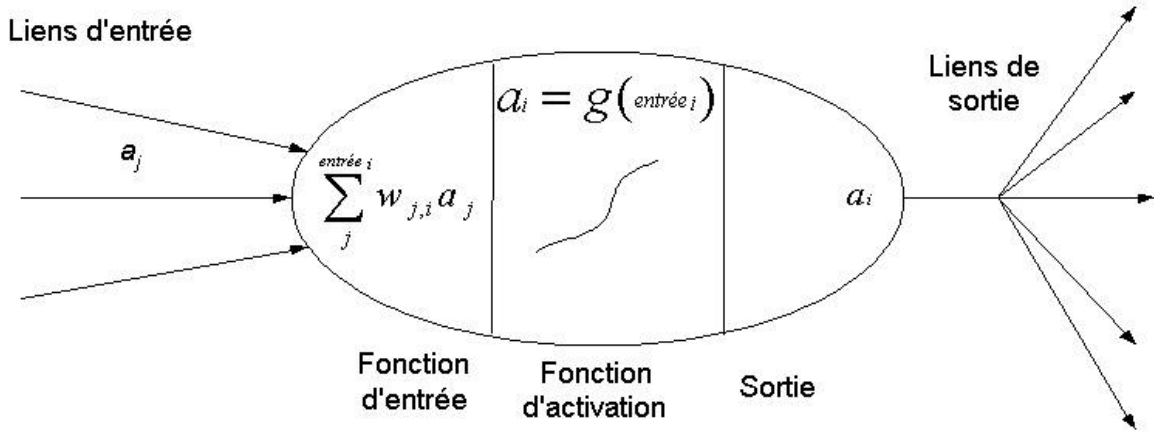


Figure 5.1 Schéma d'un neurone artificiel

D'abord, le neurone i reçoit des signaux a_j qui sont pondérés par les poids $w_{j,i}$ associés à chaque lien j et les additionnent comme l'exprime l'équation 5.1.

$$entrée_i = \sum_j w_{j,i} a_j \quad (5.1)$$

Ensuite, la valeur d'activation de l'unité i , dénotée a_i , se calcule avec l'aide de la fonction d'activation g décrite par l'équation 5.2.

$$a_i = g(entrée_i) = g\left(\sum_j w_{j,i} a_j\right) \quad (5.2)$$

Plusieurs types de fonctions d'activation peuvent être utilisés. Pour ce projet, la fonction tangente hyperbolique, décrite par la relation 5.3, est utilisée. La fonction tangente hyperbolique est définie entre les valeurs -1 et 1 .

$$g(x) = \tanh(x) = \frac{2}{1 + \exp(-2x)} - 1 \quad (5.3)$$

Finalement, l'activation a_i est propagée à l'extérieur de l'unité vers d'autres neurones pour former un réseau complet. La figure 5.2 montre le schéma d'interconnexion des unités pour les réseaux à propagation avant à une couche cachée. La couche cachée est formée des neurones qui se trouvent entre les neurones à l'entrée du réseau et ceux à sa sortie. Les connexions ne sont permises qu'entre les couches adjacentes avec toutes les unités liées entre elles de couche en couche.

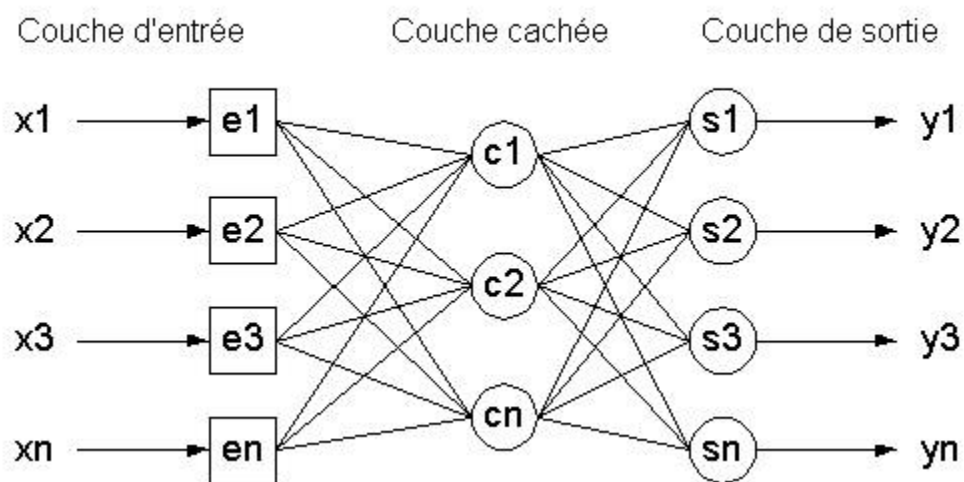


Figure 5.2 Réseau de neurones à propagation avant avec une couche cachée

La conception d'un réseau de neurones nécessite d'effectuer les choix suivants :

- Création d'un ensemble de données pour l'entraînement du réseau et d'un ensemble de données pour sa validation. Les données d'entraînement servent à montrer au réseau la fonction de correspondance qu'il doit arriver à représenter à partir des poids entre les unités. Pour un réseau qui utilise la rétro-propagation comme loi d'apprentissage, les données d'entraînement consistent en un ensemble de vecteurs qui regroupent les valeurs des entrées à soumettre au réseau et les valeurs de la couche de sorties que le réseau doit apprendre à calculer. Ces données servent pour l'apprentissage des poids du réseau. Les données de validation ont la même structure, mais ne servent pas à entraîner

le réseau : elles sont plutôt utilisées pour vérifier que le réseau arrive à bien traiter des données en entrée, non soumises lors de l'entraînement.

- Choix de la représentation des données. Le nombre d'unités d'entrées et d'unités de sorties du réseau dépendent de la représentation des données à traiter.
- Choix de la loi d'apprentissage. La loi d'apprentissage dicte comment les poids du réseau sont modifiés itérativement pour arriver à fournir les sorties désirées pour tous les vecteurs soumis lors de l'entraînement. La loi d'apprentissage utilisée dans nos travaux est la rétro-propagation standard couplée à la variante Delta-bar-Delta [HAYKIN 1999] qui vient adapter automatiquement le rythme d'apprentissage. Dans notre cas, les poids sont ajustés pour chaque vecteur d'entrée à chaque itération, ce qui constitue un apprentissage en mode continu.
- Détermination du nombre d'unités cachées. Ce sont les unités cachées qui permettent de trouver une façon appropriée de représenter la fonction de correspondance à apprendre par le réseau. Plus il y en a, plus la fonction de correspondance apprise peut être complexe. Par contre, le réseau peut aussi apprendre à ne reconnaître que les données d'entraînement. L'objectif est donc de minimiser le nombre d'unités cachées. Ceci se trouve en effectuant des essais avec différents nombre d'unités cachées sur le réseau.
- Initialisation du réseau. Lors de la création du réseau, les poids sont initialisés de façon aléatoire, sur l'intervalle $[-1,1]$ dans notre cas.

Une fois ces choix déterminés, l'entraînement du réseau peut être réalisé. Les poids du réseau sont alors modifiés selon la loi d'apprentissage et la présentation des données d'entraînement. La modification des poids s'effectue jusqu'à ce que l'erreur quadratique moyenne du réseau se soit stabilisée à sa valeur minimale. Une fois les poids stabilisés, les performances du réseau peuvent être évaluées à l'aide des données de validation.

5.2 Représentation des symboles

Les caractéristiques du symbole à reconnaître sont représentées par des valeurs réelles dans un vecteur de dimension fixe soumis en entrée du réseau de neurones. Deux méthodes pour encoder les symboles sont étudiées pour notre projet : le dimensionnement et la DCT.

5.2.1 Le dimensionnement (*scaling*)

Cette représentation consiste tout simplement à diminuer ou à agrandir la dimension du symbole perçu. Pour les symboles, il suffit de prendre la région de couleur noire et la redimensionner en une matrice 9 x 13, comme le montre la figure 5.3. Les lignes de la matrice obtenue sont alors mises bout à bout pour former le vecteur de caractéristiques servant d'entrée au réseau de neurones.

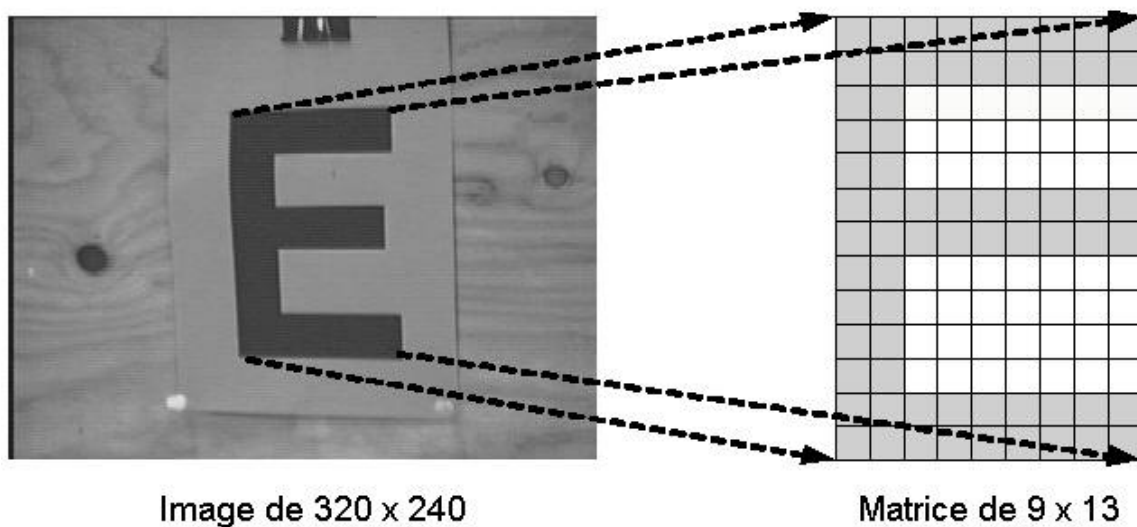


Figure 5.3 Dimensionnement du caractère E

Les dimensions de la matrice ont été fixées pour que le symbole redimensionné soit identifiable par l'humain : si un humain peut arriver à reconnaître le symbole redimensionné, nous supposons que le robot devrait lui aussi être en mesure de le faire. Les

dimensions impaires aident à centrer facilement le symbole autour du point central de la matrice, soit (5,7). Lorsque le dimensionnement du symbole est complété, toutes les valeurs comprises dans la matrice 9 x 13 sont mises bout-à-bout à partir de la ligne du haut jusqu'à la dernière ligne, pour former un vecteur d'entrée défini sur 117 unités. Les entrées dans la matrice sont aussi modifiées de façon à être définies dans l'intervalle [-1,1] : les pixels noirs prennent la valeur de (+1), et les autres, qui forment la couleur de fond, prennent la valeur de (-1).

5.2.2 La DCT

La DCT, ou transformée en cosinus, permet de représenter une image de dimension $m \times n$ en une transformée de dimension égale, mais qui représente la somme pondérée de cosinus à différentes fréquences horizontales et verticales. Cette technique est très utilisée dans les codeurs d'image ou vidéo de types JPEG et MPEG [BOVIK 2000]. La transformée en cosinus est semblable à la transformée de Fourier, car elle transpose le domaine temporel dans le domaine fréquentiel. Par contre, contrairement à la transformée de Fourier, la DCT comporte seulement des coefficients réels.

La transformée en cosinus en deux dimensions est définie par la relation 5.4.

$$H(u, v) = \frac{2}{\sqrt{MN}} C(u) C(v) \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} h(x, y) \cos \left[\frac{(2x+1)u\pi}{2M} \right] \cos \left[\frac{(2y+1)v\pi}{2N} \right] \quad (5.4)$$

La transformée inverse (IDCT) est exprimée par la relation 5.5 et est souvent utile pour vérifier si l'implémentation logicielle de la DCT est correcte. Dans ce cas, la transformée inverse (IDCT) des coefficients obtenus par la DCT doivent redonner l'image originale.

$$h(x, y) = \frac{2}{\sqrt{MN}} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} C(u)C(v)H(u, v) \cos\left[\frac{(2x+1)u\pi}{2M}\right] \cos\left[\frac{(2y+1)v\pi}{2N}\right] \quad (5.5)$$

$$C(\lambda) = \begin{cases} \frac{1}{\sqrt{2}} & \text{pour } \lambda = 0 \\ 1 & \text{pour } \lambda > 0 \end{cases}$$

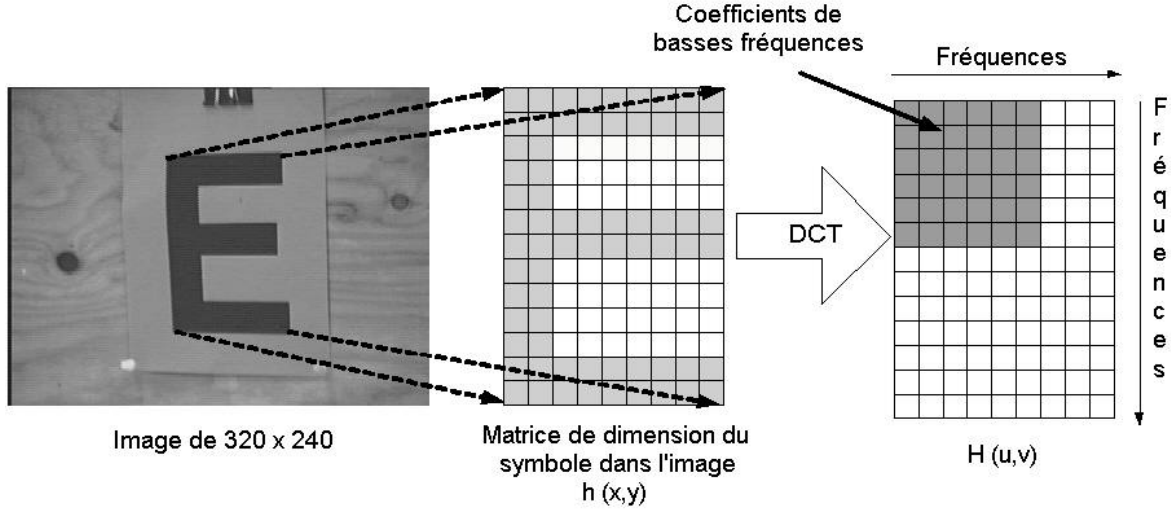


Figure 5.4 Représentation graphique de la transformée en cosinus (DCT)

Les vecteurs d'entrées sont formés des coefficients basses fréquences qui se trouvent dans le coin supérieur gauche de la matrice résultante, comme montré à la figure 5.4. Dans notre cas, l'image qui doit être transformée dépend de la grosseur du symbole dans l'image. En effet, la DCT est effectuée sur la région de couleur noire, de dimension variable, extraite par l'algorithme de segmentation des couleurs. La figure 5.4 montre un symbole de dimension 9 x 13 où sont conservés seulement 36 coefficients (6 x 6) de basses fréquences. La matrice qui représente l'image à l'entrée de la DCT est composée de (0), qui déterminent la couleur de fond (ou aucun signal), et de (1) qui composent le symbole à reconnaître de couleur noire. Contrairement au dimensionnement, la valeur (0) pour la

couleur fond est choisie au lieu de (-1) pour ne pas affecter les coefficients de basses fréquences qui sont liés à la transformée du signal d'entrée lors du calcul de la DCT.

Le nombre de coefficients de basses fréquences conservés après la transformation par la DCT est choisi de manière à conserver 80% de l'énergie totale dans les coefficients de basses fréquences. Conserver 80% de l'énergie est suffisant pour reconstituer les caractéristiques principales des symboles. L'énergie totale (E) de l'image est calculée (théorème de Parseval) à partir des coefficients de la DCT par la relation 5.6.

$$E = \sum_{u=0}^{M-1} \sum_{v=0}^{N-1} |H(u,v)|^2 \quad (5.6)$$

Le ratio d'énergie est calculé en faisant le rapport entre l'énergie contenue dans la matrice où l'on conserve les coefficients de la DCT par l'énergie totale de l'image. Après vérification à partir de toutes les images utilisées pour l'entraînement du réseau de neurones, l'énergie de tous les symboles est conservée à plus de 80% si les coefficients basses fréquences gardés ont la dimension 9×9 . Ainsi, les vecteurs de caractéristiques pour l'entraînement des réseaux de neurones sont formés par la concaténation des lignes de la matrice des coefficients de basses fréquences de la DCT de 9×9 pour obtenir des vecteurs de longueur 81.

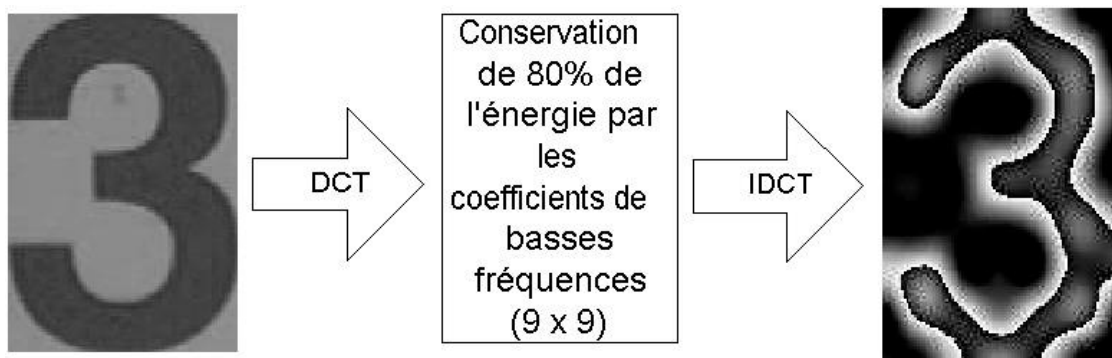


Figure 5.5 Reconstitution d'un symbole à partir des coefficients de basses fréquences de dimension 9×9

La figure 5.5 montre la reconstitution du symbole « 3 » à partir des coefficients de basses fréquences de dimension 9 x 9. Dans ce cas, nous pouvons voir que conserver environ 80% de l'énergie avec les coefficients de 9 x 9 nous donne une image très semblable au symbole original. L'idée principale de l'algorithme pour encoder les symboles par DCT est de ne pas utiliser 100% des coefficients pour limiter la taille mémoire et les calculs nécessaires pour l'entraînement et la reconnaissance par les réseaux de neurones. Par ailleurs, il serait possible d'utiliser moins de coefficients de la DCT (donc moins d'énergie) pour représenter les symboles et obtenir de bons résultats.

5.3 Performances de reconnaissance de symboles

L'entraînement des réseaux de neurones s'est effectué à partir de 35 images de chacun des 25 symboles choisis, pour un total de 875 images. L'ensemble de validation comprend 10 images de chacun des symboles, pour un total de 250 images. La couche de sortie des réseaux de neurones validés est composée de 25 unités, soit une unité pour chacun des symboles pouvant être reconnus. Pour l'entraînement, le vecteur de sortie pour un symbole donné est construit en plaçant une valeur de (+1) pour l'unité de sortie associée au symbole, et les autres à (-1). Un symbole est considéré reconnu par le réseau lorsque l'activation de l'unité de sortie associée au symbole est supérieure à 0.8 et qu'aucune autre unité a une activation supérieure à 0. L'entraînement se termine lorsque l'erreur à la sortie du réseau atteint son minimum et qu'il n'est plus possible d'améliorer les résultats d'entraînement.

Les tableaux 5.1 et 5.2 présentent les résultats calculés en effectuant la moyenne sur cinq entraînements pour chaque configuration de réseau. Plus il y a de neurones sur la couche cachée, plus grand est le nombre de poids du réseau à calculer, ce qui nécessite plus de temps de traitement. Ainsi, le nombre de neurones dans la couche cachée est augmenté

graduellement de façon à trouver la configuration qui produit peu ou pas d'erreurs sur les ensembles d'entraînement et de validation. Le nombre d'erreurs possibles est fonction du nombre d'images dans les ensembles d'entraînement et de validation et du nombre de symboles. Dans le cas de l'ensemble d'entraînement, le nombre d'erreurs possibles est 25 symboles par 25 erreurs (toutes les sorties) par 35 images, soit 21875. Le nombre d'erreurs possibles pour la validation est de 25 symboles par 25 erreurs par 10 images, soit 6250.

Le tableau 5.1 présente les résultats observés pour des réseaux de neurones utilisant la représentation de dimensionnement 9 x 13 comme entrées, pour différents nombres d'unités cachées. Au niveau des performances sur l'ensemble d'entraînement et l'ensemble de validation, c'est la configuration #10 avec 11 unités cachées qui donne les meilleures performances.

TABLEAU 5.1 Évaluation de configurations de réseaux encodés par dimensionnement

<i>Configuration</i>	<i>Nombre de neurones couche cachée</i>	<i>Nombre de poids à calculer</i>	<i>Nombre d'erreurs entraînement</i>	<i>Nombre d'erreurs validation</i>
1	2	284	248	115
2	3	426	9.6	27.6
3	4	568	0.2	23.4
4	5	710	0	15.8
5	6	852	0.2	9.2
6	7	994	0	5.4
7	8	1136	0	3.4
8	9	1278	0	3.8
9	10	1420	0	6.2
10	11	1562	0	1.8
11	12	1704	0	3.2
12	13	1846	0	4
13	14	1988	0	5
14	15	2130	0	2.8
15*	5 + 6 + 7	2556	4	9
16*	Perceptron sans couche cachée	2925 (117 * 25)	0	9

Les configuration #15 et #16 sont des cas particuliers de l'étude. La configuration #15 consiste à utiliser trois réseaux différents ayant un petit nombre de neurones sur la couche cachée et ensuite accepter une reconnaissance si deux des trois sorties obtenues sont les mêmes. L'objectif était de voir si une telle configuration permettrait d'obtenir une plus grande tolérance au bruit. Le tableau 5.1 montre toutefois que cette configuration n'est pas idéale puisqu'elle ne donne pas de meilleurs résultats que la configuration #10, et nécessite plus de calculs pour les poids. La configuration #16 consiste à entraîner un réseau par symbole en utilisant un réseau de type Perceptron (sans couche cachée). Pour cette configuration, chaque réseau a été entraîné pour reconnaître un seul symbole et à ne pas reconnaître les 24 autres. Cette configuration demande de calculer presque le double du nombre de poids de la configuration #10 et n'offre aucun avantage par rapport à celle-ci. Par conséquent, la configuration #10 avec 11 unités cachées est celle retenue.

TABLEAU 5.2 Évaluation de configurations de réseaux encodés pour la DCT

<i>Configuration</i>	<i>Nombre de neurones couche cachée</i>	<i>Nombre de poids à calculer</i>	<i>Nombre d'erreurs entraînement</i>	<i>Nombre d'erreurs validation</i>
1	1	106	996	282
2	2	212	621.4	127.8
3	3	318	401.6	121.8
4	4	424	197.6	64.2
5	5	530	0	0.6
6	6	636	0.4	0.4
7	7	742	0	0
8	8	848	0	0.4
9	9	954	0	0.4
10	10	1060	0	2.2
11	11	1166	0	0.6
12	12	1272	0	2
13	13	1378	0	0.2
14	14	1484	0	0.4
15	15	1590	0	0.4

Le tableau 5.2 présente les résultats pour des réseaux utilisant la représentation par DCT des symboles, en conservant les coefficients de basses fréquences de dimension 9×9 . La configuration #7, avec 7 neurones sur la couche cachée, est celle retenue, car aucune erreur d'entraînement ni de validation est présente.

6 RÉSULTATS

Les chapitres 3, 4 et 5 ont présenté les caractéristiques et les performances pour chacune des trois étapes requises pour notre approche d'interprétation visuelle de symboles par un robot mobile. Il est maintenant possible d'évaluer les performances pour l'ensemble de ces étapes. La section 6.1 traite des résultats obtenus en laboratoire sous des conditions d'éclairage contrôlées, tandis que la section 6.2 présente les performances de l'approche dans des environnements courants.

6.1 Expérimentations avec éclairage contrôlé

Au laboratoire, il nous est possible de réaliser des essais sous des conditions d'éclairage uniformes et contrôlables. Deux conditions d'éclairage ont été utilisées : l'éclairage par fluorescents (normal), et l'éclairage par lumières encastrées (faible).

Les expérimentations réalisées consistent à placer un robot dans un enclos de 12' x 12'. Des symboles furent placés sur les murs de l'enclos de manière à obtenir un symbole par mur, localisé au centre de celui-ci. Tous les 25 symboles ont été placés tour à tour au même endroit dans l'enclos pour vérifier les performances de l'approche développée. Le robot se déplaçait librement dans l'enclos, évitant les murs au besoin, prenant l'image d'un symbole et, après un délai de 10 secondes, répétait le même processus. Ainsi, les images des symboles prises par le robot furent obtenues sous différents angles et distances.

Le robot traita 25 lectures de symbole pour chacun des symboles spécifiés au chapitre 2, pour chacune des trois couleurs de fond (bleu, rose et orange) et dans les deux conditions d'éclairage. Au total, 1250 images de symboles furent traitées par le robot lors de ces tests.

Les résultats obtenus sont présentés selon l'encodage utilisé pour l'interprétation des symboles par le réseau de neurones, soient le dimensionnement et la DCT.

Les tableaux 6.1 et 6.2 présentent les performances de reconnaissance pour l'ensemble des symboles. Une reconnaissance est réalisée lorsque la neurone de sortie du réseau ayant l'activation la plus grande est supérieur à 0.8 et que ce neurone est associé au symbole à reconnaître. Dans ces tableaux, la distinction est faite entre les symboles non reconnus (c'est-à-dire lorsque le réseau de neurones n'a pas d'unité à la sortie ayant une activation plus grande que 0.8) et les symboles qui donnent une reconnaissance incorrecte (c'est-à-dire que le neurone de sortie qui donne la valeur la plus élevée ne correspond pas au symbole perçu). Ainsi, il est préférable que le taux de reconnaissances incorrectes soit faible. En effet, dans le cas d'un symbole non reconnu, le robot ne ferait aucune action. Cependant, dans le cas d'une reconnaissance incorrecte, le robot pourrait effectuer des actions qui ne seraient pas appropriées.

TABLEAU 6.1 Taux de reconnaissance pour l'encodage par dimensionnement

Couleur de fond	Éclairage	Reconnu (%)	Non reconnu (%)	Incorrecte (%)
Bleu	Fluorescent	88.31	5.42	6.27
Bleu	Encastré	94.69	3.05	2.25
Rose	Fluorescent	89.46	8.07	2.47
Rose	Encastré	91.47	5.31	3.22
Orange	Fluorescent	89.89	5.62	4.49
Orange	Encastré	93.26	4.65	2.09

TABLEAU 6.2 Taux de reconnaissance pour l'encodage par DCT

Couleur de fond	Éclairage	Reconnu (%)	Non reconnu (%)	Incorrecte (%)
Bleu	Fluorescent	83.39	7.73	8.88
Bleu	Encastré	93.60	4.64	1.76
Rose	Fluorescent	88.94	8.09	2.97
Rose	Encastré	89.98	7.92	2.10
Orange	Fluorescent	87.80	7.54	4.65
Orange	Encastré	91.47	7.25	1.29

Les résultats montrent que le taux de reconnaissance est meilleur quand l'éclairage est plus faible. Ceci s'explique par le fait que les reflets des fluorescents sur le symbole de couleur noir provoquent parfois des distorsions assez grandes pouvant mener à une non reconnaissance ou une reconnaissance incorrecte. En moyenne, 91.2% des symboles sont bien reconnus avec l'encodage par dimensionnement, et 89.2% avec l'encodage par DCT. La couleur de fond des symboles affecte légèrement les performances de reconnaissance, ce qui est dû au fait que les canaux de couleurs sont entraînés manuellement et que la perception des couleurs par l'humain peut varier lors de l'entraînement. Aussi, certaines couleurs absorbent mieux la lumière, ce qui crée moins de distorsions dans les symboles.

Le tableau 6.3 montre les taux de reconnaissance obtenus pour chacun des symboles. Dans la plupart des cas, les symboles sont mieux reconnus lorsque l'éclairage est encastré. Les matrices de confusion pour l'encodage par dimensionnement et par DCT sont présentées aux annexes 1 et 2. Ces matrices de confusions sont obtenues en faisant la moyenne des taux de reconnaissance avec l'éclairage encastré et fluorescent pour chaque couleur de fond. Les diagonales des matrices de confusion représentent les taux des symboles qui sont bien reconnus. La dernière colonne de chaque matrice de confusion montre les taux des symboles non reconnus. À partir des annexes 1, 2 et du tableau 6.3, il est facile de vérifier

que les symboles qui ont un faible pourcentage de reconnaissance (comme par exemple 0, 9, W et L) sont souvent non reconnus et ne sont pas pris pour d'autres symboles. Le problème réside donc dans l'extraction des couleurs et non dans la reconnaissance des symboles. Par contre, avec le dimensionnement, il y a confusion entre le 3 et le 8. Avec la DCT, une confusion survient entre le 7 et le 9. De plus, le symbole W est difficile à identifier avec la DCT, probablement parce que la variation d'éclairage a une grande influence sur les coefficients de basses fréquences pour ce symbole.

TABLEAU 6.3 Taux de reconnaissance de chacun des symboles

Symbole	Taux avec dimensionnement		Taux avec DCT	
	Fluorescent	Encastré	Fluorescent	Encastré
0	74.67	93.33	73.33	94.74
1	85.33	90.67	94.59	92.00
2	94.67	96.00	89.19	90.67
3	73.33	89.33	84.00	92.00
4	88.73	89.33	84.51	92.00
5	96.00	98.65	79.45	93.33
6	98.63	93.33	84.93	90.79
7	96.00	86.30	94.59	100.00
8	86.67	96.00	90.67	95.95
9	60.00	94.67	73.33	87.84
A	86.67	94.52	92.00	97.30
C	100.00	100.00	88.00	93.33
E	89.33	96.00	90.54	94.67
H	87.50	77.03	87.72	85.14
J	98.67	94.67	98.67	100.00
L	88.00	90.67	73.33	63.51
N	74.32	82.43	100.00	97.30
S	95.89	100.00	93.15	100.00
V	90.67	93.24	98.67	94.59
W	84.72	88.00	40.28	68.49
↑	98.67	98.67	96.00	98.65
↓	100.00	100.00	89.33	98.67
←	89.33	90.67	85.33	89.47
→	93.33	94.59	85.33	85.14
↗	98.67	100.00	100.00	96.00

Tous les résultats présentés jusqu'à maintenant sont basés sur le fait que les symboles sont agrandis pour obtenir la résolution maximale de ceux-ci. Cependant, dans le but d'accélérer le temps de traitement de notre système, il peut être intéressant de déterminer la résolution minimale requise pour interpréter visuellement un symbole. À cette fin, les symboles ont été placés directement en face du robot à une distance de huit pieds, permettant ainsi de valider la reconnaissance des symboles à partir de la résolution minimum jusqu'au maximum de résolution. La résolution d'un symbole est mesurée en fonction du nombre de pixels qu'il représente en hauteur et en largeur dans l'image. La résolution minimale est choisie à partir de la matrice 9 x 13 utilisée pour le dimensionnement du symbole. Ainsi, une résolution de 9 x 13 signifie que la correspondance entre l'image du symbole perçu et l'entrée du réseau de neurones est identique. La résolution maximale dépend de la forme du symbole. En effet, tel que décrit au chapitre 4, la caméra est positionnée pour centrer le centre de gravité du symbole dans l'image tout en laissant un espace tout autour du symbole. Les symboles qui ont un centre d'aire très bas (comme la lettre L) ne peuvent pas être agrandis autant que les symboles symétriques (comme le chiffre 8).

Pour prendre les images à différentes résolutions, les valeurs de zoom sont augmentées graduellement jusqu'à ce que la hauteur ou la largeur du symbole perçu dépasse un certain facteur entier qui multiplie les dimensions 9 x 13 de référence.

TABLEAU 6.4 Taux de reconnaissance des symboles pour différentes résolutions

Résolution	Facteur de résolution	Taux de reconnaissance avec dimensionnement (%)	Taux de reconnaissance Avec DCT (%)
9 x 13	1	58.4	11.2
18 x 26	2	84.2	9.6
27 x 39	3	91.2	23.2
36 x 52	4	94.4	44.8
45 x 65	5	96.8	63.2
54 x 78	6	99.2	79.2
63 x 91	7	99.2	84.0
72 x 104	8	100.0	95.2
81 x 117	9	97.6	99.2
90 x 130	10	99.2	99.2
99 x 143	11	100.0	98.4
108 x 156	12	98.4	97.6

Le tableau 6.4 présente les résultats moyens obtenus en utilisant cinq images de chaque symbole pour chaque résolution. Au total, 1500 images ont été prises. Les résultats obtenus indiquent clairement que les taux de reconnaissance de la représentation avec dimensionnement sont meilleurs que ceux avec la DCT. La résolution de 54 x 78 semble suffisante pour reconnaître 99.2% des symboles encodés par dimensionnement alors qu'il faut une résolution de 81 x 117 pour obtenir les mêmes résultats avec l'encodage par DCT. Les taux de reconnaissance supérieurs à la moyenne des taux présentés dans les tableaux 6.1 et 6.2 sont dus au fait que les images se trouvent directement en face de la caméra et ne comportent pas de distorsion angulaire.

Le tableau 6.5 présente les nouveaux résultats pour le temps de capture optimisé en utilisant un facteur de résolution égal à 6. Ces résultats sont obtenus en plaçant un symbole devant le robot en faisant la moyenne des temps de capture de 25 essais à la même distance du robot.

Ainsi, le temps de capture moyen est de 10.3 secondes, un gain de 37% par rapport au temps de capture avec la résolution maximale.

TABLEAU 6.5 Temps de capture optimisés à différentes distances

Distance (pieds)	Temps (secondes)
2	5.5
3	7.1
4	7.9
5	8.4
6	9.6
7	11.5
8	13.4
9	13.2
10	16.2

6.2 Expérimentations dans des conditions d'éclairage non-contrôlées

Les tests réalisés dans des conditions d'éclairage non-contrôlées permettent de vérifier si les algorithmes proposés fonctionnent adéquatement dans des conditions d'illumination diverses. Pour ces tests, l'encodage des symboles fut celle par dimensionnement et les symboles agrandis à la résolution maximale. La couleur de fond des symboles fut l'orange et les expérimentations ont eu lieu au Centre des congrès de Austin, Texas, lors de la conférence AAAI 2000. Le robot naviguait en utilisant des symboles pour se guider (par les flèches par exemple) ou comme points de repères pour l'ascenseur, les salles de conférence, la station de recharge, etc. L'éclairage variait de façon importante puisque certains corridors étaient très bien éclairés avec de grandes fenêtres, tandis que d'autres endroits comme l'ascenseur, étaient très sombres. Les salles de conférence étaient très bien éclairées, un peu comme les conditions en laboratoire.

Dans les endroits sombres, l'extraction du symbole par les couleurs était possible seulement en inclinant les symboles vers le haut pour mieux les éclairer. Il fallait aussi faire attention pour ne pas placer les symboles dans les endroits trop éclairés, puisque les réflexions de la lumière (souvent du soleil) amenaient des distorsions dans la perception des symboles. Comme anticipé, l'ajustement des couleurs avec l'interface graphique décrite à la section 3.3 fut nécessaire et s'avéra un outil indispensable pour la réalisation des expériences dans un environnement inconnu. Dans ces conditions, le robot a été en mesure d'identifier 83% des symboles dans l'environnement avec 17% des symboles non-reconnus et aucune reconnaissance incorrecte.

Les limites de l'approche en utilisant les couleurs résident donc dans le fait que le robot doit être en mesure de bien percevoir les symboles, dans des conditions d'éclairage variées, avant de les reconnaître. La variation des couleurs selon l'éclairage étant très importante, l'approche décrite pour ce mémoire fonctionne bien dans le cas où l'éclairage est suffisamment uniforme, puisqu'il n'est pas possible d'entraîner toutes les variations des couleurs dans la même table. L'utilisation de plusieurs tables d'appartenance des couleurs est donc requise dans des conditions d'éclairage variées. À partir de l'intensité moyenne des pixels de l'image, qui serait une bonne indication de la luminosité de l'endroit où se trouve le robot, il serait possible d'utiliser la bonne table d'appartenance afin de percevoir les couleurs correctement. Cette hypothèse n'a pas été vérifiée pour le présent ouvrage et devrait l'être dans les travaux futurs pour s'assurer que le système de vision fonctionne bien dans différentes conditions d'opération.

7 DISCUSSION

Donner la capacité de lire des symboles serait certainement utile à un robot mobile autonome. Il fut déjà identifié au chapitre 1 que la localisation du robot par rapport à une carte pourrait bénéficier de symboles comme repères visuels. L'identification d'objets par un symbole peut aussi être utile. Un objet à transporter pourrait être étiqueté par un symbole et le robot pourrait prendre cet objet parmi plusieurs qui sont étiquetés différemment. Cette idée intéressante est déjà utilisée par un hélicoptère autonome miniature pour l'identification de matériel dangereux qu'il doit transporter [AMIDI 1996]. Un symbole peut aussi servir à identifier la présence d'une station de recharge, ce qui permet à un robot mobile de prendre les actions nécessaires (c'est-à-dire tourner de 180 degrés) afin de détecter la station et de s'y brancher [MICHAUD et coll. 2000], comme le montre la figure 7.1. Cette habileté fut en fait validée en utilisant les travaux du présent mémoire.

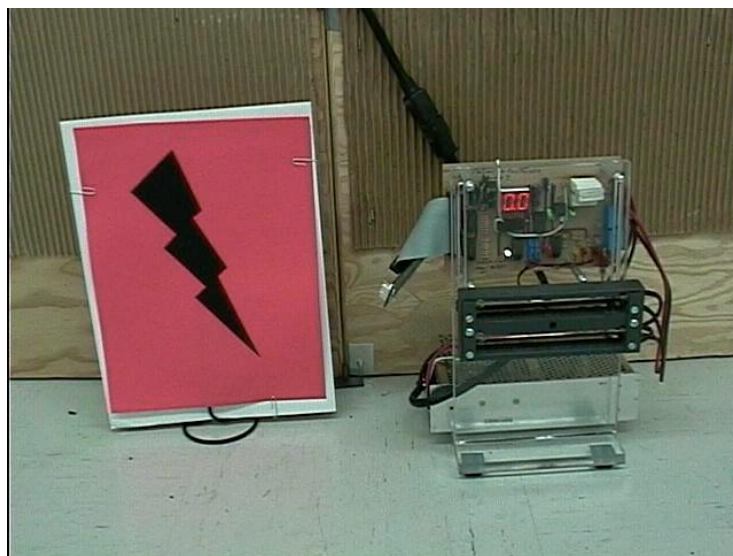


Figure 7.1 Identification d'une station de recharge par un symbole

Pour revenir aux explications présentées au chapitre 1 concernant la nécessité de repérer d'autres agents à proximité et de communiquer entre eux, des symboles pourraient être placés sur la structure des robots pour les identifier. Cette hypothèse n'est pas valide pour reconnaître des humains ou des animaux. En contre-partie, les êtres vivants n'utilisent pas un seul moyen pour y arriver et de la même façon, l'approche peut être vue comme complémentaire et couplée à d'autres comme la reconnaissance de paroles, de visages, de formes ou d'autres moyens électroniques. La reconnaissance mutuelle des robots permettrait une communication plus élaborée par la suite, soit par la vision, par radio, ou par tout autre moyen de communication. En installant un écran sur les robots, il serait aussi possible qu'ils puissent « lire » et « écrire » afin de communiquer de façon dynamique des renseignements concernant leurs états internes ou des connaissances particulières qu'ils ont acquis durant leurs opérations.

Enfin, au lieu de traiter un seul symbole à la fois, une approche plus générale pourrait être développée afin de permettre à des robots de reconnaître des mots et même des phrases. Des modules comportementaux pourraient être ajoutés pour extraire la position des mots dans l'image et imiter la lecture de gauche à droite des humains. De cette façon, l'identification des numéros de porte permettrait à un robot comme Xavier [SIMMONS et coll. 1997] de livrer du courrier sans se fier uniquement à sa carte interne utilisée pour la localisation. De plus, il serait possible d'identifier les véhicules, par leurs plaques d'immatriculation, à même les véhicules en mouvement et non par des caméras fixes [HERMIDA et coll. 1997][TINDALL 1997]. Pour des messages plus diversifiés, des techniques de traitement du langage naturel pourraient aussi être utilisées. Les travaux du présent ouvrage démontrent la faisabilité de donner la capacité à des robots d'interpréter

visuellement des symboles et s'avèrent un premier pas dans l'élaboration de stratégies plus complexes pour la lecture de symboles.

CONCLUSION

Ce mémoire présente une approche permettant à un robot mobile, doté d'une caméra couleur Pan-Tilt-Zoom et d'un ordinateur embarqué, d'identifier des symboles dans des environnements réels. L'approche consiste à extraire un symbole à partir d'une image segmentée par les couleurs. Les informations tirées de cette segmentation permettent de signaler la présence d'un symbole pour que le robot s'immobilise et positionne sa caméra avec un contrôleur PID pour le contrôle du Pan et du Tilt et une heuristique pour le contrôle du Zoom visant à obtenir une image du symbole avec une résolution adéquate pour la reconnaissance. La reconnaissance s'effectue par un réseau de neurones artificiels préalablement entraîné par rétro-propagation, avec des images de symboles obtenues par le robot.

L'approche fonctionne pour des symboles formés d'un seul segment, placés perpendiculairement au sol, pour des symboles d'une certaine couleur (noir dans notre cas) complètement entourés par une autre couleur (bleu, rose ou orange dans notre cas). Les meilleures performances sont obtenues en utilisant un réseau de neurones à trois couches, 11 unités cachées et en utilisant l'encodage par dimensionnement. Des performances de 89.2% en éclairage fluorescent et de 93.1% en éclairage encastré sont observées sur l'ensemble de 25 symboles. Le temps de traitement varie de 8 à 27 secondes pour des distances de 2 à 10 pieds en utilisant la résolution maximale des symboles perçus. Les temps de traitement peuvent être optimisés en agrandissant les symboles à percevoir à une résolution de 54 x 78. Dans ce cas, les temps de traitement sont de 5 à 16 secondes (un gain de 37%).

L'approche décrite dans ce mémoire est extensible et peut être utilisée pour reconnaître d'autres symboles de différentes couleurs. La couleur des symboles et la couleur de fond doivent être choisies de manière à être facilement identifiables par rapport à l'environnement. Des couleurs voyantes se trouvant rarement dans l'environnement facilitent le positionnement de la caméra et l'extraction du symbole dans l'image. Une fois les couleurs choisies pour le fond et le symbole lui-même, il faut ajouter les couleurs dans la table d'appartenance avec différentes conditions d'éclairage, tel que décrit dans le chapitre 3, pour être capable d'extraire les symboles de l'image. Pour ajouter des symboles à reconnaître, il suffit de créer des ensembles d'entraînement et de validation comprenant chacun une version des symboles à reconnaître. Dans le cas de ce mémoire, les ensembles d'entraînement et de validation comprenaient chacun des 25 symboles pris sous différents angles. Les symboles doivent être sélectionnés de façon à éviter les confusions entre les symboles qui se ressemblent (5 et S, I et 1, etc.) et doivent comprendre un seul segment pour éviter de recombinaison des segments pour former les symboles entiers. Enfin, lorsque les ensembles d'entraînement et de validation sont complétés, il s'agit de trouver la configuration de réseau de neurones avec un nombre minimal de neurones dans la couche cachée pour obtenir de bons résultats (peu ou pas d'erreurs) sur les ensembles d'entraînement et de validation, comme montré au chapitre 5.

Finalement, les travaux présentés dans ce mémoire démontrent avec succès la faisabilité de mettre en œuvre des capacités d'interprétation visuelle de symboles sur des robots mobiles. Les résultats obtenus démontrent que la technique peut fonctionner dans des environnements où l'éclairage est plutôt uniforme. Par contre, des algorithmes plus sophistiqués pour l'extraction des couleurs devront être développés pour mieux compenser

l'effet de l'éclairage sur la perception des couleurs. D'autres techniques comme l'extraction des contours ou des textures pourraient être évaluées lors de travaux futurs. De plus, ces travaux peuvent donc servir de base pour le développement d'approches plus sophistiquées permettant la reconnaissance de caractères manuscrits, tels que réalisés par des systèmes de reconnaissance commerciaux ou de symboles comprenant plusieurs segments (ou parties) qu'il faut recombinaison. Cependant, il faut adapter ces techniques pour qu'elles puissent fonctionner en temps réel sur un robot mobile opérant dans des environnements courants. Avec l'avancement de la technologie, la puissance de calcul des robots devrait continuer à croître de manière exponentielle avec les années, ce qui permettra de concevoir des algorithmes de reconnaissance de plus en plus complexes et performants. Nos machines intelligentes, robotiques ou autres, pourront alors bénéficier d'informations transmises par écrit, ce moyen créé par l'humain il y a de cela plusieurs milliers d'années et qui nous a permis d'évoluer de génération en génération.

BIBLIOGRAPHIE

AMIDI, O. (1996) An Autonomous Vision-Guided Helicopter, Thèse de doctorat, Robotic Institute, Carnegie Mellon University.

ARKIN, R.C. (1992) *Cooperation without communication: multi-agent schema based robot navigation*, Journal of Robotic Systems, vol. 9, n°3, p. 351-364.

ARKIN, R.C., BALCH, T. (1998) *Cooperative multiagent robotic systems*, dans D. Kortenkamp, R.P. Bonasso et R. Murphy, éditeurs, Artificial Intelligence and Mobile Robots : Case Studies of Successful Robot Systems, AAAI/MIT Press, Cambridge, MA, p. 277-296.

BALCH, T., BOONE, G., COLLINS, T., FORBES, H., MACKENZIE, D., SANTAMARIA, J. (1995) *Io, Ganymede and Callisto - A multiagent robot trash-collecting team*, AI Magazine, vol. 16, n°2, p. 39-51.

BECKERS, R., HOLLAND, O.E., DENEUBOURG, J.L. (1994) *From local actions to global tasks: Stigmergy and collective robotics*, dans R. Brooks et P. Maes, éditeurs, Artificial Life IV, Proc. Fourth International Workshop on the Synthesis and Simulation of Living Systems, MIT Press, p. 181-189.

BOVIK, A. (2000) Handbook of Image & Video Processing, Academic Press, 891 p.

BROOKS, R.A. (1986) *A robust layered control system for a mobile robot*, IEEE Journal of Robotics and Automation, RA-2(1), p. 14-23.

BRUCE, J., BALCH, T. VELOSO, M. (2000) *Fast and cheap color vision on commodity hardware*, dans Proc. Workshop on Interactive Robotics and Entertainment, Pittsburgh. p. 11-16.

BUNKE, H., WANG, P.S.P. (1997) Handbook of Character Recognition and Document Image Analysis, World Scientific, 852 p.

CAO, Y., FUKUNAGA, A.S., KAHNG, A.B., MENG, F. (1997) *Cooperative mobile robotics: antecedents and directions*, Autonomous Robots, vol. 4, p. 1-23.

DAUTENHAHN, K. (1999) *Embodiment and interaction in socially intelligent life-like agents*, Artificial Intelligence, vol. 1562, p. 105-147.

DUDEK, G., JENKIN, M., MILIOS, E., WILKES, D. (1995) *Experiments in sensing and communication for robot convoy navigation*, dans Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems, p. 268-273.

DUDEK, G., JENKIN, M., MILIOS, E., WILKES, D. (1993) *A taxonomy for swarm robots*, dans Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems, p. 441-447.

EVERETT, H.R. (1995) Sensors for Mobile Robots, Theory and Application, Wellesley, Massachusetts, A K Peters, 200 p.

HAYKIN, S. (1999) Neural Networks – A Comprehensive Foundation, 2^e édition, Prentice Hall, 842 p.

HERMIDA, X.F., RODRIGUEZ, F.M., LIJO, J.L.F., SANDE, F.P. INLESIAS, M.P. (1997) *An O.C.R. for V.L.P.'s (Vehicule License Plates)*, dans Proc. International Conference on Signal Processing Applications & Techonolgies 1997.

JUNG, D., HEINZMANN, J., ZELINSKY, A. (1998) *Range and pose estimation for visual servoing on a mobile robotic target*, dans Proc. IEEE International Conference on Robotics and Automation, vol. 2, p. 1226-1231.

LITTLE, J. (1998) *Robot Partners: Collaborative Perceptual Robotic Systems*, UBC, <http://www.cs.ubc.ca/spider/little/links/robuds.html>.

LORIGO, L.M., BROOKS, R.A., GRIMSON, W.E.L. (1997) *Visually guided obstacle avoidance in unstructured environments*, dans Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems, p. 373-379.

MATARIC, M.J. (1994) Interaction and Intelligent Behavior, Thèse de doctorat, Department of Electrical Engineering and Computer Science, MIT AI Lab (MIT AI Lab AI-TR 1495).

MATARIC, M.J. (1993) *Kin recognition, similarity, and group behavior*, dans Proc. Fifteenth Annual Cognitive Science Society Conference, Boulder, Colorado, p. 705-710.

MAXWELL, B.A., MEEDEN, L., ADDO, N., BROWN, L., DICKSON, P. NG, J., OLSHFSKI, S., SILK, E, WALES, J. (2001) *Alfred : the robot waiter who remembers you*, Autonomous Robot, à paraître.

MCLURKIN, J. (2000) *Swarm, distributed programming of autonomous robots*, <http://research.isr.com/swarm/index.htm>.

MICHAUD, F. AUDET, J., LÉTOURNEAU, D., LUSSIER, L., THEBERGE-TURMEL, C., CARON, S. (2001) *Experiences with an autonomous robot attending the AAAI Conference*, accepté pour présentation dans IEEE Intelligent Systems.

MICHAUD, F., LETOURNEAU, D. (2001) *Mobile robot that can read symbols*, dans Proc. CIRA 2001, Banff, p. 338-343.

MICHAUD, F., LETOURNEAU, D. (2001) *Teaching a robot how to read symbols*, dans Proc. Autonomous Agents 2001, Montréal, p. 184-185.

MICHAUD, F., AUDET, J., LETOURNEAU, D. (2000) *Having a robot attend AAAI 2000*, IEEE Intelligent Systems, vol. 15, n°6, p. 74-76.

MICHAUD, F., LETOURNEAU, D., AUDET, J., BELANGER, F. (2000) *Symbol recognition and artificial emotion for making an autonomous robot attend the AAAI Conference*, dans Proc. AAAI Conference, Austin, Texas, p. 1140-1141.

MICHAUD, F., VU, M.T. (1999) *Managing robot autonomy and interactivity using motives and visual communication*, dans Proc. International Conference on Autonomous Agents, Seattle, WA, p. 160-167.

MITRA, S. (2000), Digital Signal Processing, A Computer Based Approach, 2^e édition McGraw-Hill, 888 p.

MORAVEC, H. (1999) Robot: Mere Machine to Transcendent Mind, Oxford Press, 224 p.

OGATA, K. (1997) Modern Control Engineering, 3^e édition, Prentice Hall, 997 p.

PARKER, L.E. (1994) Heterogeneous Multi-robot Cooperation, Thèse de doctorat, MIT AI Lab (MIT AI Lab AI-TR 1465).

PERZANOWSKI, D., ADAMS, W., SCHULTZ, A.C., MARSH, E. (2000) *Towards seamless integration in a multi-modal interface*, dans Proc. Workshop on Interactive Robotics and Entertainment, Pittsburgh, p. 3-9.

NOTAKE, Y., KATO, K., YAMAMOTO, K. (2000) *Recognition of characters distorted by camera angle*, dans Proc. Vision Interface, Montréal, p. 274-279.

RESNIK, M. (1991) Turtles, Termites, and Traffic Jams: Explorations in Massively Parallel Microworlds, MIT Press, 184 p.

REKLEITIS, I. DUDEK, G., EVANGELOS, M. (1995) *Multi-robot exploration of an unknown environment efficiently reducing the odometry error*, dans Proc. International Joint Conference in Artificial Intelligence, vol. 2, 1997, p. 1340-1345.

ROWLEY, H., BALUIA, S., KANADE, T. (1998) *Rotation invariant neural network-based face detection*, dans Proc. IEEE Conference on Computer Vision and Pattern Recognition, p. 38-44.

SCHNEIDERMAN, H. KANADE, T. (2000) *A statistical model for 3D object detection applied to faces and cars*, dans Proc. IEEE Conference on Computer Vision and Pattern Recognition.

SIM, T. SUKTANKHAR, R. MULLIN, M., BALUJA, S. (2000) *Memory-based face recognition for visitor identification*, dans Proc. 4th International Conference on Face Gesture Recognition, p. 214-220.

SIMMONS, R.G., GODWIN, R., HAIGH, K.Z., KOENIG, S., O'SULLIVAN, J., VELOSO, M.M. (1997) *Xavier: Experience with a layered robot architecture*, ACM Magazine Intelligence, p. 22-33.

SONKA, M., HLAVAC, V. BOYLE, R. (1998), Image Processing, Analysis, and Machine Vision, 2^e édition, PWS Publishing, 828 p.

TINDALL, D.W (1997) *Deployment of automatic license plate recognition systems in multinational environments*, European Conference on Security and Detection, publication no. 437.

VELOSO, M., WINNER, E., LENSER, S., BRUCE, J., BALCH, T. (2000) *Vision-servoed localization and behavior-based control for an autonomous quadruped legged robot*, dans Proc. Workshop on Interactive Robotics and Entertainment, Pittsburgh. p. 95-101.

VU, M.T. (2000) Communication visuelle par signalement lumineux avec un robot mobile, Mémoire de maîtrise, Département de génie électrique et de génie informatique, Université de Sherbrooke.

		Symbole reconnu																									
		0	1	2	3	4	5	6	7	8	9	A	C	↗	E	H	J	L	←	N	→	S	↑	V	W	non	
S y m	0	84.0%	0.0%	0.0%	0.0%	0.0%	2.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.7%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	13.3%	
	1	0.7%	87.9%	0.7%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	2.7%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	2.7%	5.4%	
	2	0.0%	0.0%	95.3%	2.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.7%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	2.0%	
	3	0.0%	0.0%	0.0%	80.4%	0.0%	0.0%	0.0%	10.8%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.7%	0.7%	0.0%	7.4%	
b o	4	0.0%	0.0%	0.0%	0.0%	89.1%	0.0%	0.0%	0.0%	0.0%	0.7%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	1.4%	0.0%	0.0%	0.0%	0.0%	8.7%	
	5	0.0%	0.0%	0.0%	0.0%	0.0%	96.6%	0.7%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.7%	0.0%	0.0%	0.0%	0.7%	0.7%	0.0%	0.0%	0.0%	0.0%	0.7%		
	6	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	95.3%	0.0%	2.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	2.7%		
	7	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	91.2%	0.0%	0.0%	0.0%	0.0%	0.0%	2.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.7%	6.1%	
e	8	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	91.2%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	8.8%	
	9	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	77.1%	0.0%	0.0%	0.0%	0.7%	3.4%	0.0%	0.0%	0.0%	4.0%	0.0%	0.0%	0.0%	0.0%	14.8%		
	A	0.0%	0.7%	0.0%	0.0%	0.7%	0.0%	0.0%	0.0%	0.0%	0.0%	90.5%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	8.1%		
	C	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	100%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%		
r e	↘	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	100%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%		
	E	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	100%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%		
	C	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.7%	0.0%	0.7%	0.0%	0.0%	0.0%	0.0%	0.0%	94.6%	0.0%	0.0%	0.7%	0.0%	0.0%	0.0%	0.0%	0.7%	2.0%		
	H	0.0%	0.0%	0.0%	0.0%	1.0%	0.0%	3.8%	0.0%	0.7%	0.0%	0.0%	0.0%	0.0%	0.0%	5.3%	77.5%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	11.6%		
o n	J	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	96.6%	0.0%	0.7%	0.0%	0.0%	0.0%	0.0%	0.0%	2.7%		
	L	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	1.3%	0.0%	0.7%	0.0%	0.0%	89.3%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	8.7%		
	←	0.7%	0.0%	0.0%	0.0%	0.0%	0.0%	3.4%	0.0%	0.0%	0.7%	0.0%	0.0%	0.0%	0.0%	0.7%	0.0%	0.0%	89.9%	0.0%							

ANNEXE 2 - MATRICE DE CONFUSION POUR LES SYMBOLES ENCODÉS PAR DCT

		Symbole reconnu																										
		0	1	2	3	4	5	6	7	8	9	A	C	↗	↓	E	H	J	L	←	N	→	S	↑	V	W	non	
S y m b o l e a r r e c o n n a î t r e	0	83.3%	0.0%	1.3%	0.0%	0.0%	0.0%	0.0%	0.0%	8.0%	0.0%	0.0%	0.7%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.7%	0.0%	0.7%	5.3%	
	1	0.0%	93.3%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	6.7%	
	2	0.0%	0.0%	89.9%	3.4%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	2.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.7%	0.0%	0.0%	4.1%	
	3	0.0%	0.0%	0.7%	86.4%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.7%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	12.2%	
	4	0.0%	0.0%	0.0%	0.0%	86.4%	0.0%	0.0%	0.0%	0.0%	0.0%	4.1%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	1.3%	0.0%	0.0%	0.0%	6.2%	
	5	0.0%	4.1%	0.0%	0.0%	0.0%	85.6%	2.8%	0.0%	0.0%	0.7%	0.7%	0.0%	0.0%	0.0%	0.0%	0.7%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	5.4%	
	6	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	87.9%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.7%	0.0%	0.0%	0.0%	0.0%	0.7%	0.0%	0.0%	0.0%	10.8%		
	7	0.0%	1.4%	1.4%	0.0%	0.0%	0.0%	0.0%	0.0%	97.3%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	
	8	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.7%	0.0%	0.0%	92.6%	0.0%	0.0%	0.7%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	6.0%	
	9	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	11.3%	0.0%	80.5%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	8.1%	
	A	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	94.6%	0.7%	0.0%	1.4%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	3.4%	
	C	0.0%	0.0%	0.7%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	90.7%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	8.7%	
	e	↗	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	98.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	2.0%	
	↘	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	94.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	5.3%	0.0%	0.0%	0.7%	
	E	0.0%	0.0%	0.7%	0.0%	0.0%	0.0%	0.0%	0.7%	0.0%	0.0%	0.7%	0.0%	0.0%	0.0%	0.0%	92.6%	0.0%	0.0%	0.0%	0.7%	0.0%	0.0%	0.0%	0.0%	0.0%	4.7%	
	O	H	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.7%	0.0%	0.0%	0.7%	0.0%	1.0%	0.0%	0.0%	83.8%	2.8%	0.0%	0.0%	0.0%	1.0%	0.0%	0.0%	0.0%	0.0%	10.0%	
n	J	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	99.3%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.7%		
a	L	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	8.7%	0.0%	1.3%	0.0%	0.0%	0.0%	67.8%	0.0%	2.0%	0.0%	0.0%	0.0%	0.0%	20.1%		
î	←	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.7%	0.0%	0.0%	0.0%	0.0%	5.3%	0.0%	0.0%	85.2%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	8.8%		
r	N	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.7%	0.0%	0.0%	0.0%	0.0%	98.6%	0.0%	0.0%	0.0%	0.0%	0.0%	0.7%		
t	→	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	2.7%	0.0%	0.0%	0.0%	84.5%	0.7%	0.0%	0.0%	0.0%	12.2%		
r	↑	0.0%	0.7%	0.7%	0.0%	0.0%	0.0%	0.7%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.7%	0.0%	0.0%	0.0%	96.6%	0.0%	0.0%	0.0%	0.7%		
e	↗	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	97.3%	0.0%	0.0%	0.0%	2.7%		
V	↓	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.7%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	96.0%	0.0%	0.0%	3.4%		
W	↘	0.0%	0.0%	0.7%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.7%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	96.0%	0.0%	44.1%		